ATARI

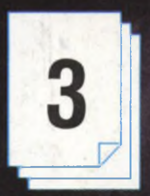
ST COMPUTER

Die Fachzeitschrift für den ATARI-ST Anwender.

März '87

ÖS 53,—/Sfr. 6,—

DM 6,—



Das Sprachgenie

Alle Programmier-sprachen für den ST

Was die Zukunft bringt ...

Die ST Computer im Gespräch mit ATARI Deutschland

Tabellenkalkulation mit der Maus

VIP Professional im GEM Gewand

Die ST-Ecke

Tips für den ST-User

NEU!



Viel Neues...

...gibt es in diesem Monat zu berichten. Nachdem der Anfang des Jahres überstanden ist, haben auch wir unsere guten Vorsätze in die Tat umgesetzt. Wir haben uns entschlossen, einige Dinge an unserer Zeitschrift zu ändern. Neben dem neuen, hoffentlich ansprechenden Layout haben wir zwei neue Rubriken aufgenommen: „Die Musikseiten" werden sich mit der Musikalität des ST beschäftigen. Dabei wollen wir nicht nur Musikprogramme wie unseren Musik-Editor, MUSIX32 oder andere vorstellen – auch die Keyboard-Begeisterten sollen auf ihre Kosten kommen. Wir nehmen an, daß auf dem MIDI-Markt noch einige Neuigkeiten zu erwarten sind und wollen Sie auf dem Laufenden halten.

Die zweite Rubrik haben wir eingerichtet, da uns immer wieder Fragen erreichen, deren Antworten für alle Leser interessant sein dürften. Die „ST-ECKE" soll eine TIPS&TRICKS-Kolumne sein, in der wir kleinere (oder auch mal größere) Probleme aufgreifen und eine Lösungsmöglichkeit anbieten, die meist sogar als Programm ausformuliert sein wird. Ohne Ihre rege Teilnahme sind wir natürlich alleingelassen. Deshalb: Wenn Sie fragen in Bezug auf Programmentwicklung (AES, VDI, BIOS, XBIOS etc.) haben, dann schreiben Sie uns bitte!

Obwohl die Zeitschrift umfangreicher geworden ist, sind noch mehr Farbseiten hinzugekommen. Wir sind auch ein wenig stolz darauf, daß seit Bestehen der ST-COMPUTER die Anzahl der Seiten immer gewachsen ist, ohne daß sich der Preis geändert hat. Wir wollen dafür sorgen, daß dies auch weiterhin so bleibt.

Neues gibt es auch bei ATARI. Wieder wurden die Preise gesenkt. Es scheint, als wolle man nun einen Preisspielraum für die nachfolgenden Geräte schaffen. Endlich gibt es die neuen Geräte im PC-Gehäuse. Gerade im professionellen Bereich öffnen sich damit vielleicht die Türen für so manchen ATARI-Computer, da man nun nicht mehr für jedes einzelne Gerät einen eigenen Netzstecker benötigt – Kabelsalat ade. Bliebe noch zu wünschen, daß auch der Adress- und Datenbus vollständig herausgeführt und somit das System noch offener wird. Besonders Entwickler können sich auf die 2- und 4-Megabyte-Versionen freuen, die vielleicht sogar mit Blitter ausgeliefert werden!? Einen weiteren Coup hat Jack Tramiel mit der Einführung des ATARI-MS-DOS-Rechners gelandet. Nicht nur, daß er sicher wieder ein gutes Geschäft wittert, der Einstieg in die MS-DOS-Welt könnte zudem Atari – immer noch mit Videospielen in Verbindung gebracht – ein neues, professionelles Image verleihen.

Ob Neues nur angekündigt wird oder auch in absehbarer Zeit erscheint, das wird uns nicht bloß die CeBIT zeigen, auf die ATARI und sicher auch einige Softwarehäuser hinarbeiten. Oft genug wurden die großen Ankündigungen der Softwareanbieter ein Flop. Hoffen wir, daß die Programme, die ab diesem Jahr erscheinen, nicht bloß Umsetzungen für den ST sein werden, sondern für ihn speziell entwickelt werden, und daß sich die Softwarehäuser bei der Entwicklung Zeit lassen – nichts schadet dem Image eines Computers mehr als unausgereifte Software.

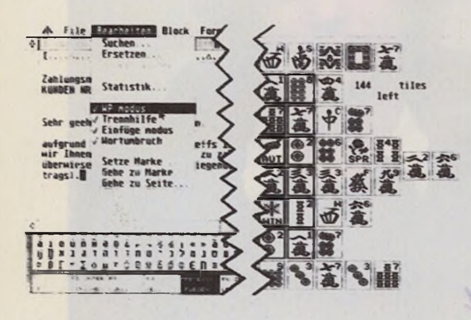
Wir wünschen Ihnen einen interessanten Besuch auf der CEBIT. Wenn Ihnen die Zeit für einen Messebesuch fehlt, vertrauen Sie uns: Wir werden in der nächsten Ausgabe alles Wichtige zusammentragen und berichten.

Stefan Höhn
(freier Mitarbeiter der Redaktion)

Das Sprachgenie

Inzwischen gibt es eine ganze Menge leistungsstarker Programmiersprachen. Über ihre Herkunft, den Einsatzbereich und die verschiedenen ST-Versionen berichtet dieser Artikel.





Die zwei Gesichter des ST

K-Switch macht aus einem Rechner mit einem MegaByte Speicher kurzerhand zwei mit je 512 KByte.
Umschalten per Tastendruck: Ideal, um die Arbeitszeit zu versüßen. Doch auch für sinnvolle Anwendungen ist dieses Programm geeignet.

VIP im GEM-Gewand

VIP Professionel ist sicher ein Begriff. Die neue Version hat nun ein GEM-Gewand verpaßt bekommen: eine sehr zweckmäßige Kleidung. Doch auch im Inneren hat sich einiges getan.



Messen, was meßbar ist!?

Herkömmliche Speicheroszilloskope sind in der Regel sehr teuer. Für den ST-Anwender steht eine preisgünstige Lösung zur Verfügung. Ob diese Lösung eine Alternative ist, erfahren Sie in unserem Bericht.

Elemente der Künstlichen Intelligenz

Um die neue Computersprachengeneration etwas näherzubringen, starten wir in diesem Heft eine Serie über dieses Thema. Sie basiert auf den zwei Public-Domain-Sprachen XLISP und PROLOG 10. Ein Blick in diese Serie ist sicher interessant.



Messe mit Gewinn
Die CES in Las Vegas

Um Ihnen alle Neuigkeiten auf dem ST-Markt präsentieren zu können, schickten wir unseren rasenden Reporter in die weite Welt.

Allgemeines

Software

Hardware

Grundlagen

Kurse

Aktuelles

News

Bei ATARI purzeln die Preise!

Zum Anfang dieses Jahres wurden die Preise bei ATARI teilweise erheblich gesenkt. Die neuen Preise:

1040 ST/F DM 1998,–
(Rechner mit eingebauter SF 314)

520 ST/M DM 998,–
(Rechner m. eingebaut. HF-Modulator)

SM 124 DM 498,–
(Monochrom-Monitor)

SF 314 DM 598,–
(doppelseitiges Laufwerk)

Seit Februar '87 kostet die 20 MByte Festplatte SH 204 nur noch DM 1298,–!!!

Damit haben, wie ATARI mitteilt, die Preise eine Untergrenze erreicht. Weitere Preissenkungen sind deshalb nicht zu erwarten.



Nun doch: 1st Word Plus!

Nach langem Warten und unserer Meldung im letzten Heft, daß 1st Word Plus gestoppt sei, ist es jetzt doch lieferbar. Es wird zusammen mit dem Serienbriefprogramm 1st_Mail ausgeliefert. Die neuen Features gegenüber der bisherigen Version wurden ja schon kurz beschrieben. In der nächsten Ausgabe wird ein ausführlicher Test folgen. Der Preis beträgt DM 199,–.

ATARI Corp. (Deutschland) GmbH
Frankfurter Str. 89–91
6096 Raunheim

Neues von MICRODEAL

Die englische Softwarefirma MICRODEAL hat zwei neue Programme herausgebracht. Das erste ist eine Druckerutility namens MI-PRINT und dient



u. a. zum Ausdruck formatierter Listings.

Das zweite Programm heißt **THE SPRITE CONSTRUCTION KIT** und dient, wie auch nicht anders zu erwarten, dem Erstellen von Sprites. Es arbeitet vollständig GEM-unterstützt und läuft nur in niedriger Auflösung. Sprites lassen sich mit bis zu 16 Farben definieren. Außerdem ist es relativ einfach, mit den erstellten Sprites eine Animation zu erzeugen, das Programm ermöglicht einen Testlauf der Animation. Zum „gleichzeitigen" Bearbeiten von verschiedenen Sprites dient ein Clipboard (Pinwand), an dem die jeweiligen Sprites zwischengespeichert werden können. Ferner ist eine Ausschneide- und Einfüge-Routine vorhanden, mit der es möglich ist, ganze Spriteteile auszutauschen.

Die Preise und der Vertrieb sind uns zur Zeit leider noch nicht bekannt.

Ein Programm für den Amateurfunker

Die Buchhandlung Franke bietet ein sehr umfangreiches, komfortables Amateurfunk-Fernschreibprogramm für alle ST-Rechner an. **Radio-Writer ST** ist vollständig in GEM eingebunden, läßt sich aber auch über die Tastatur steuern und läuft in hoher und mittlerer Auflösung. Selbstverständlich wird neben dem Programm noch ein beliebiger Fernschreibkonverter mit AFSK (optional mit Sende-/Empfangsrelais für KOX) für die serielle Schnittstelle benötigt. Folgende Besonderheiten zeichnen das Programm aus:

- vollautomatische Sendersteuerung (KOX)
- eingebauter Druckerpuffer (Spooler)
- Betrieb mit BAUDOT-, ASCII- und ATARI Binär-Code
- alle wichtigen Geschwindigkeiten und Zeichenformate einstellbar
- geteilter Bildschirm für Senden und Empfangen mit Wort- und Zeilenkorrektur
- integrierter Selektivruf-Auswerter mit Logbuch
- 'Unshift on Space', Auto-Linefeed, Wortumbruch
- abspeicherbare Geräteeinstellungen
- fünf frei programmierbare Standardtexte
- Datei-Ein- und Ausgabe auch während laufender Übertragung

Radio-Writer ST kostet DM 98,–.

Buchhandlung Franke
Steiner Straße 5
7531 Eisingen
Tel. 07232/81664

Alte Drucker aufgemöbelt

Die Besitzer alter EPSON-Drucker der Serien RX, FX, MX und JX haben Anlaß zur Freude. Die Firma Weide Elektronik bietet ein NLQ-Modul an, das auch bei diesen älteren Druckern ein Schriftbild in Near-Letter-Qualität erzeugt. Es muß lediglich ein EPROM im Drucker ausgetauscht werden, jeder kann diese Umrüstung leicht vornehmen. Verschiedene Schriftarten lassen sich durch eine bestimmte Tastenfolge der eingebauten Druckertasten anwählen. Das NLQ-Modul kostet DM 149,–.

Weide-Elektronik
Regerstraße 34
4010 Hilden

Hier ein kleiner Probeausdruck, der mit einem RX 80 F/T erstellt wurde.

Volks-Adress

Eine schnelle Adressverwaltung wird mit dem Programm Volks-Adress für DM 69,– angeboten. Es sucht nach allen eingegebenen Kriterien. Drei Felder lassen sich frei belegen (z. B. mit einer persönlichen Notiz oder dem Geburtstag). Ferner läßt sich die Druckerausgabe beliebig aus den verwendeten Feldern definieren, so daß man zum Beispiel nur die Daten für Adressetiketten ausdrucken kann. Zusätzlich kann man weitere Informationen wie „DRUCKSACHE" mitdrucken lassen. Das definierte Druckformat läßt sich abspeichern und später wiederverwenden. Bis zu 12 Sonderzeichen können an den Drucker angepaßt werden.

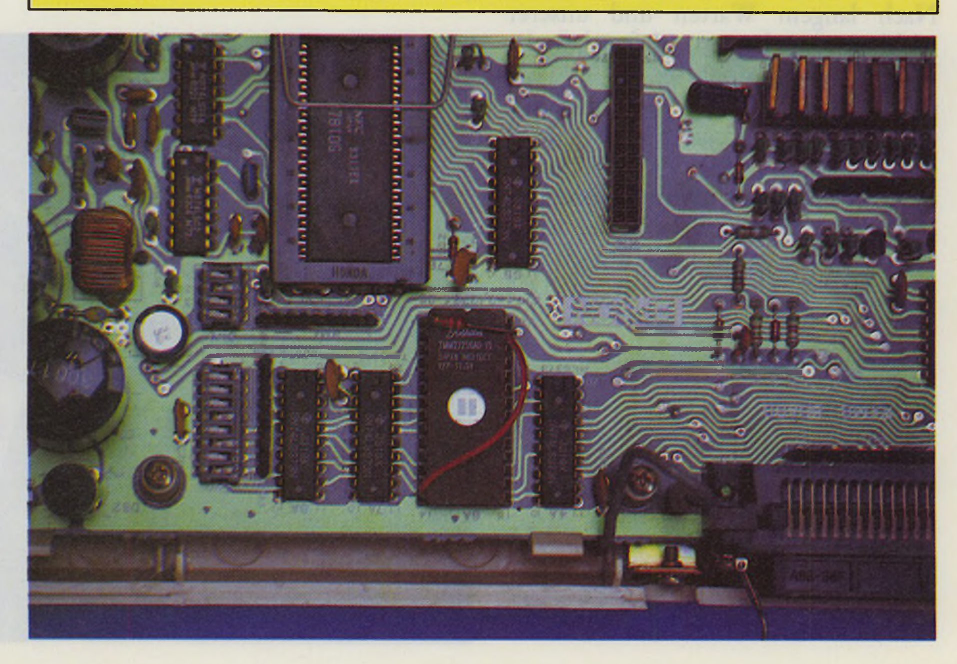
Eine weitere Option dient der Serienbrieferstellung mit 1st_Word oder 1st_Word Plus. Dazu muß man zuvor alle in Frage kommenden Adressen auswählen. Die Anrede wird automatisch (je nachdem, wie man sie in der Adressmaske bestimmt hat) in den Brief eingesetzt.

Ulrich Kort
Pappelweg 6
3030 Walsrode 1
Tel. 05161/1618

Sehr geehrte Damen und Herren,

aus Anlaß der bevorstehenden Markteinführung
unserem neuesten Produkt bekanntmachen.

Unter dem Namen "***NLQ-MODUL***" bieten wir ab so
satz für eine Reihe Epson-Drucker aus den Ser
JX an. Durch das ***NLQ-MODUL*** ergeben sich i



TiM (Time is Money)

Mit TiM bietet die C.A.S.H. GmbH ein vollständig GEM-unterstütztes Buchführungsprogramm für ST-Serie an. TiM wendet sich an alle, die jederzeit einen Überblick über ihre finanzielle Lage wünschen, und setzt keinerlei Kenntnis über Buchführung voraus. Benötigt wird ein monochromer Monitor und mindestens ein Laufwerk. TiM unterstützt Festplattenbetrieb und alle gängigen Drucker. Das 80seitige Handbuch ist in Einführung, Referenz und Musterbuchungen untergliedert. Der Preis beträgt DM 198,–. Das Handbuch kann im voraus für DM 25,– bestellt werden, die beim Kauf von TiM angerechnet werden. Eine Demo-Version ist gegen einen Unkostenbeitrag von DM 7,– erhältlich.

C.A.S.H. GmbH
Schillerstr. 64
8900 Augsburg
Tel. 0 82 37 / 10 20

DABAS V.1.0

Die Vertriebsgesellschaft Rausch & Haub bietet ab Anfang März ein neues Datenbanksystem für den ATARI ST an. DABAS arbeitet GEM-unterstützt und hat folgende Features:

- frei wählbare Dateimaske
- logische Verknüpfungen beim Suchen
- beliebig tiefer Sortierpfad
- Serienbriefe mit 1st_Word, Übersichten, Etiketten, Listendruck
- Dateigröße unabhängig vom RAM
- deutsches Handbuch

Vorgesehene Erweiterungen: Rechenfunktionen, Grafikverwaltung, Fakturierung, Kalkulation etc.

DABAS benötigt mindestens 1 MByte RAM, 1–2 Laufwerke, optional Harddisk und Drucker, sowie einen monochromen Monitor.

Rausch & Haub GbR
Berliner Freiheit 16
5300 Bonn 1
Tel. 02 28 / 63 83 13

Referenzkarte zum GFA BASIC-Interpreter

Ab sofort kann jeder GFA BASIC-Interpreter-Besitzer, der seine ausgefüllte Registrierkarte eingeschickt hat, eine kostenlose Referenzkarte bei der GFA-Systemtechnik GmbH anfordern. Diese Referenzkarte enthält die Befehle des Interpreters nach Gruppen übersichtlich geordnet. Alle Interessenten schicken bitte folgendes zusätzlich mit:

- Name und vollständige Anschrift
- Seriennummer des Interpreters
- einen frankierten, an sich selbst adressierten C5-Rückumschlag

GFA Systemtechnik GmbH
Heerdter Sandweg 30
4000 Düsseldorf 11
Tel. 02 11 / 58 80 11

Molekülstrukturen auf dem ST

Zu dem in den NEWS der Januar-Ausgabe vorgestellten Programm ATOMIUM gibt es nun eine verbesserte Version. Unter anderem wurde der Moleküleditor stark verbessert. Ferner ist jetzt möglich, ein Molekül in sämtlichen Richtungen zu drehen, um somit einen Eindruck von der jeweiligen Struktur zu bekommen.

Alle, die bereits die erste Version erworben haben, können gegen Einsendung der Original-Diskette und eines frankierten Rückumschlages die neue Version erhalten.

Der Preis beträgt unverändert DM 149,–.

Michael Schaumburg
Halemweg 21
1000 Berlin 13
Tel. 0 30 / 382 31 05

ST Pascal < > OSS Pascal

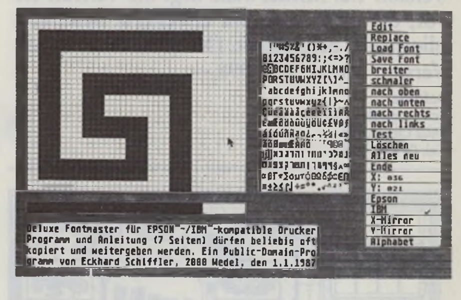
Bei dem OSS-Pascal handelt es sich um die für den ausländischen Markt bestimmte Version des ST-Pascals von CCD. Der Lieferumfang unterscheidet sich jedoch. Bei den in Deutschland erhältlichen OSS-Pascal-Versionen handelt es sich um sogenannte Reimporte, bei denen keinerlei Support von CCD gewährleistet ist.

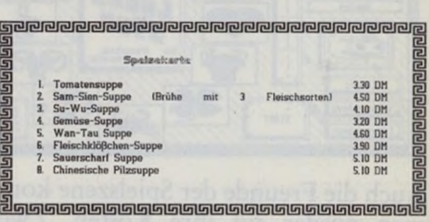
CCD
Burgstr. 9
6228 Eltville
Tel. 0 61 23 / 16 38

PD-News

Auch im Bereich der Freeware-Szene hat sich wieder einiges getan. Einige neue Programme wollen wir Ihnen kurz vorstellen.

'DELUXE-FONTMASTER' ist das Programm, das uns am meisten beeindruckte. Der Autor Eckhard Schiffler lehrte sein Werk das Drucken von verschiedenen Schriftstücken, beispielsweise Speisekarten, Einladungen und sonstigen Plakaten – und zwar mit eigenen Zeichensätzen. In einem Dokument können sieben verschiedene, übrigens sehr gelungene Zeichensätze benutzt werden. Dabei druckt DELUXE-FONTMASTER in Proportionalschrift und Blocksatz.




Speisekarte

1. Tomatensuppe 3.30 DM
2. Sam-Sinn-Suppe (Brühe mit 3 Fleischsorten) 4.50 DM
3. Su-Wu-Suppe 4.10 DM
4. Gemüse-Suppe 3.20 DM
5. Wan-Tau Suppe 4.60 DM
6. Fleischklößchen-Suppe 3.90 DM
7. Sauerscharf Suppe 5.10 DM
8. Chinesische Pilzsuppe 5.10 DM

Ganz nebenbei kann alles mit verschiedenen Rahmen versehen werden. Auch wenn Sie schöne Briefe drucken wollen, bietet sich dieses Programm an, denn es enthält einen eigenen Texteditor. Bei längeren Briefen oder Texten empfiehlt sich aber, den Text mit einem Textprogramm zu erstellen und nur den Ausdruck über den Fontmaster zu leiten. Wer gerne seine künstlerischen Fähigkeiten walten lassen möchte, der kann mit dem Fonteditor, der ebenfalls zu diesem Programm gehört, eigene Zeichensätze erstellen oder die bestehenden verändern. Das Programm unterstützt die gängigen EPSON/IBM-kompatiblen Drucker und liefert ein fast druckreifes Schriftbild (PD 40).

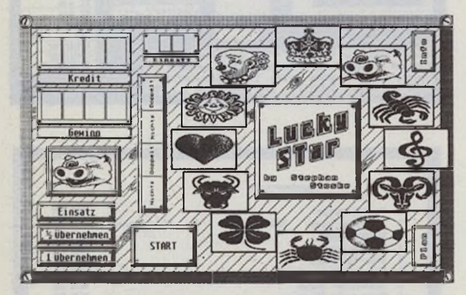
Doodle, das PD-Programm der ersten Generation, hat nun wohl endgültig zwei Nachfolger gefunden. Nach ANDROMEDA (PD 35), das wir sehr empfehlen können, erschien jetzt das Programm PAINTER von Armin Happel. Es beherrscht neben den gängigen Funktionen verschiedenen Blockoperationen (Drehen, Verschieben, Spiegeln...). Eine gute gelungene Lupe

macht Feinheiten des Bildes zugänglich. Außerdem beinhaltet PAINTER einen Füllmuster- und Linieneditor sowie Einstellmöglichkeiten für Kreis- und Ellipsenbogen. PAINTER verarbeitet diverse Bildformate (DEGAS, DOODLE) und zeigte sich, leider nur auf monochromem Bildschirm, durchaus konkurrenzfähig (PD 41).

Neue Version des Mark Williams C

Der Softwaretest des Mark Williams C-Compilers in der letzten ST Computer beruhte auf der Version 1.0. Mittlerweile liegt die Version 1.1 vor, die unter anderem erweiterte Systembibliotheken bereit stellt. Der Preis dieser neuen Version beträgt DM 398,—.

Softline
Schwarzwaldstraße 8a
7602 Oberkirch
Tel. 078 02/37 07



Auch die Freunde der Spielszene kommen wieder auf ihre Kosten. Zwei Spielautomaten, JACKPOT von Roland Hädicke und LUCKY von Stefan Stoske, verleiten zum Glücksspiel. Beide Programme stellen Automaten dar, bei denen verschieden hohe Einsätze dann vervielfältigt werden, wenn man beispielsweise zwei Schweine in Serie oder in Kombination mit einer Sonne erspielt. Leider sind auch diese Programme nur auf s/w-Monitor lauffähig, aber vielleicht ist dies auch als Revanche für jene Spielprogramme zu sehen, die ausschließlich auf Farbe laufen (PD 39).

Konstrukteuren, die sich mit statischen Berechnen beschäftigen müssen, sei ein Blick in FRAM77, das Programm von Hans Peter Wendorf empfohlen (PD 42).

Für alle Schüler und Studenten, die sich mit Darstellung und mathematischer Berechnung von Ausgleichskurven beschäftigen *'dürfen'*, hat Thomas Proffen (Wurmsoftware) ein Programm erstellt, das sicherlich eine Arbeitsersparnis bei Labor- und ähnlichen Berichten erbringt. Es arbeitet mit optionaler logarithmischer Achseneinteilung, wahlweise x-Wert-Automatik, Rastereinteilung, automatischer Achsenbeschriftung und vielem mehr (PD 42).

Damit Sie, liebe Leser, ohne große Mühe diese Software erhalten können, haben wir den Public-Domain-Service geschaffen. Ab sofort liefern wir die PD-Software auf qualitativ hochwertigen Disketten der Marke **FUJI MF1DD** (135tpi), mit denen auch wir sehr gute Erfahrungen gesammelt haben.

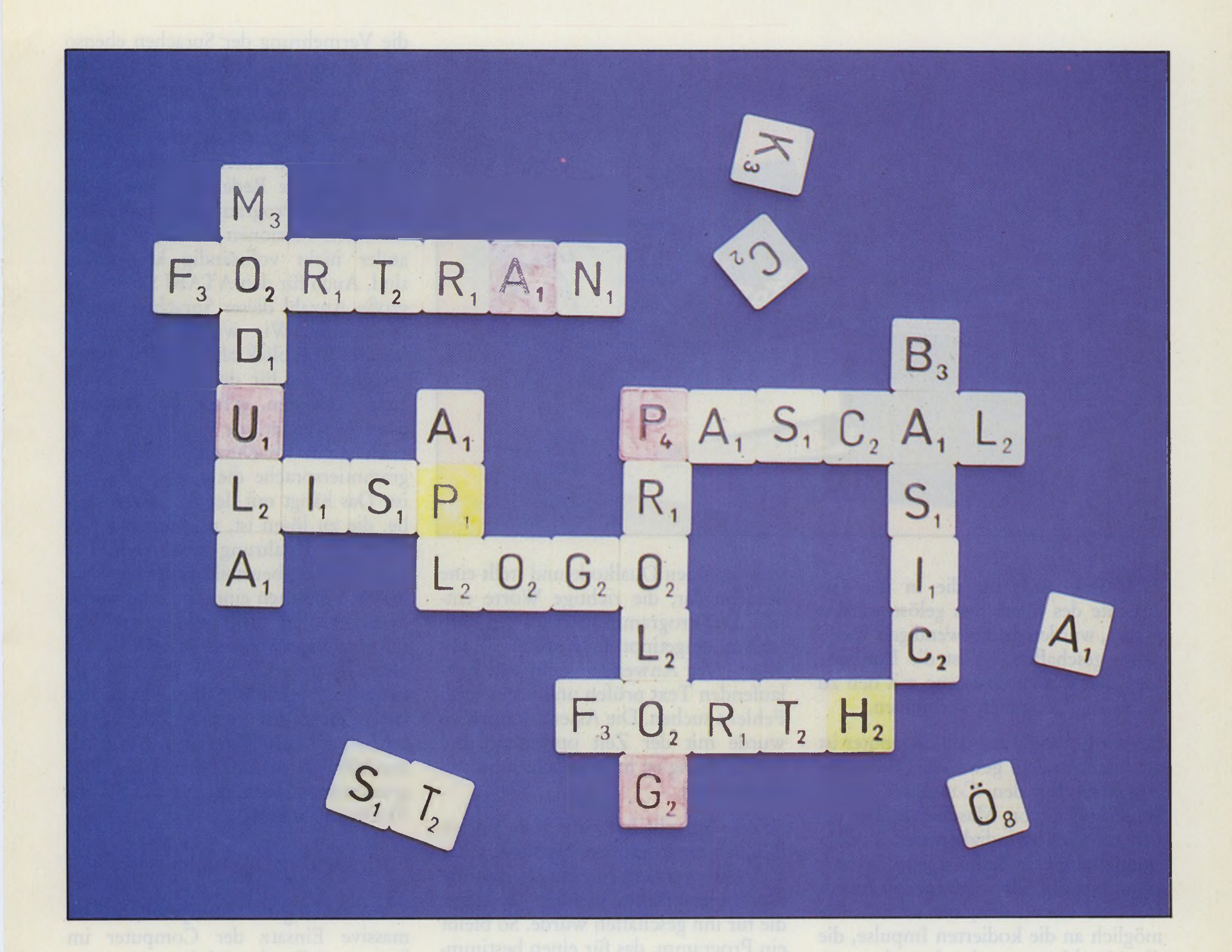
Der Preis pro bespielter PD-Diskette liegt bei DM 10,–.

Da jeder ST-Besitzer in ständigem Diskettenmangel lebt (das lehrt die Erfahrung) und unser Preis kaum über dem normalen Kaufpreis einer Diskette liegt (wenn nicht sogar darunter), glauben wir ein wirlich interessantes Angebot zu machen.

Zur Bestellung genügt ein einfacher Brief mit einem Adressrückaufkleber und einem Verrechnungsscheck über den entsprechenden Betrag. Die Lieferung erfolgt mit UPS frei Haus. Nähere Informationen entnehmen Sie bitte den gesonderten Public-Domain-Seiten.

Da es sich hierbei um einen reinen Service handelt, bitten wir Sie, sich an die Versandbedingungen zu halten. Bei steigendem Arbeitsaufwand könnten wir das Angebot nicht halten.

Bezugsadresse:
ST-Computer Redaktion
Public Domain Service
Postfach 5969
6236 Eschborn 1



Das Sprachgenie:

Alle Programmiersprachen

Wäre das Leben nicht einfacher, wenn wir alle die gleiche Sprache sprechen würden? Man könnte sogar davon träumen, daß dann alle Spannungen zwischen den Menschen verschwinden und Kriegsgefahren oder ähnliche nicht natürliche Katastrophen verbannt würden. Leider ist das nur ein Traum. Viele haben ihn geträumt. Das Esperanto sollte ein besseres Verständnis auf der Basis einer gemeinsamen Sprache schaffen. Kurz, nachdem der polnische Arzt L. Zamenhof diese universelle Sprache entwickelt hatte, wurden kulturelle Brücken zu einer Mode. Keine dieser Kunstsprachen hat sich indes durchgesetzt; die Menschheit bleibt durch verschiedene Sprachen getrennt.
Vielleicht, weil Computer ein Produkt des Menschen sind, spiegeln sich auch hier menschliche Verhaltensweisen.
Die Existenz verschiedener Computer-Sprachen beruht jedoch nicht nur auf der Tatsache, daß diese von verschiedenen Menschen geschaffen werden. Sicher hat sie mehr zu tun mit der Vielfältigkeit der Probleme, die der Mensch dem Computer gestellt hat.



Für jede Aufgabe, die in der Geschichte des Menschen gelöst werden mußte, wurden die notwendigen Werkzeuge geschaffen. So ist die Entwicklung der Wissenschaft eng mit den zu lösenden Problemen verbunden.

Bei der Entwicklung der Computer ist das nicht anders gewesen. Die ersten Aufgaben für den Computer waren Steuerungen von Maschinen, bei denen die Geschwindigkeit der Ausführung eine große Rolle spielte. So war es notwendig, eine Kommunikationsform zu entwickeln, die so nah wie möglich an die kodierten Impulse, die die Maschine versteht, herankommt. Am Anfang wurde der Rechner mechanisch eingestellt, um eine bestimmte Funktion zu realisieren. Durch die Veränderung von Reglern oder Schaltern wurde diese analytische Maschine veranlaßt, etwas zu realisisieren. Mit verschiedenen Schaltereinstellungen wurden dem Rechner verschiedene Befehle übermittelt.

John von Neuman machte eine revolutionäre Entwicklung in der Geschichte der Computer-Mensch-Kommunikation. Er gab dem Rechner keine mechanischen Instruktionen mehr, sondern symbolische: Eine schier endlose Folge von Nullen und Einsern, die die Maschine als Befehl interpretierte. Doch damit waren noch keine längeren und fehlerlosen Programme zu entwickeln.

Erst mit dem nächsten Schritt wurde ein für den Menschen bequemes und verständliches Bindeglied zur Maschine geschaffen. Die Assemblersprache befreit den Programmierer von dem verwirrenden Dualkode und stellt eine Sprache dar, die richtige Worte enthält. Das Programm wird auf den Bildschirm eingetippt und später assembliert. Der Anwender kann den fortlaufenden Text prüfen und leicht nach Fehlern suchen. Die Assemblersprache wurde mit der Zeit optimiert: ihre Handhabung ist mittlerweile recht unkompliziert.

So eine Sprache hat zwar viele Vorteile, doch einen großen Nachteil: Jeder Prozessor versteht nur eine bestimmte Assemblersprache, nämlich diejenige, die für ihn geschaffen wurde. So bleibt ein Programm, das für einen bestimmten Prozessor geschrieben wurde, für einen anderen völlig unverständlich!

Problemorientierte Sprachen

In den 50er Jahren lernte man aus den oben beschriebenen Erfahrungen und forschte nach einer besseren Kommunikationsmöglichkeit mit dem Rechner. Man wollte eine Sprache schaffen, die so weit wie möglich die Technik der menschlichen Kommunikation widerspiegelt. Die Struktur der Probleme, die an den Computer gestellt wurde, sollte dem angepaßt werden. In jedem Fall sollte es eine Sprache sein, die auf verschiedenen Computern laufen könnte.

Mit der Zeit wurde der Computer ein Polyglott. Zahlreiche Sprachen wurden für die unterschiedlichen Probleme geschaffen, so daß heute die Landschaft der Computersprachen unübersichtlich geworden ist.

Im Bereich der Personal-Computer ist die Vermehrung der Sprachen ebenso schnell vor sich gegangen wie in der Welt der größeren Anlagen. Nur wenige, völlig unbedeutsame Sprachen wurden nicht an die verschiedenen PC angepaßt. Von einem Standard kann also nicht die Rede sein. Vor allem bringt jede Programmiersprache verschiedene Versionen hervor, die zueinander nicht vollständig kompatibel sind. Auch für den ATARI ST ist eine große Anzahl dieser Sprachen bereits vorhanden. Wir wollen hier einen kurzen Überblick aller Programmiersprachen, die für den ATARI ST in Frage kommen, geben. Wir können dem Anwender die entscheidung jedoch nicht abnehmen, welche Programmiersprache die richtige für ihn ist. Das hängt mit der Art der Aufgabe, die zu lösen ist, und mit der persönlichen Erfahrung zusammen. Für eine Leistungsbeurteilung der verschiedenen Versionen einer Sprache würde dieser Artikel nicht ausreichen. Wir machen unsere Leser deshalb darauf aufmerksam, daß viele der hier genannten Programmiersprachen in früheren Ausgaben unserer Zeitung getestet und beurteilt wurden. Natürlich beschäftigen wir uns nur mit den Programmiersprachen, die für den ST zur Verfügung stehen.

Fortran: Der Großvater

Am Anfang der 50er Jahre, als der massive Einsatz der Computer im kommerziellen Bereich begann, waren die Kosten der Software-Entwicklung enorm gestiegen. Die meiste Zeit wurde für die Korrektur von Fehlern aufgewandt, die sich im Programm eingeschlichen hatten. 1954 gelang es IBM, die erste höhere Programmiersprache zu schaffen. FORTRAN (Formula Translator) ist eine Programmiersprache, die von Anfang an zur Lösung von naturwissenschaftlichen Problemen gedacht war. Da zu diesem Zeitpunkt die meisten Aufgaben der Datenverarbeitung Probleme aus dem mathematischen Bereich waren, hatte man mit FORTRAN eine Sprache gefunden, die die Entwicklung und Lösung komplizierter arithmetischer und algebraischer Operationen beherrschte. Der komplizierte Umgang mit der Ein- und Ausgabe von Daten war damals kein Hindernis, diese Programmiersprache in großem Umfang einzusetzen. Da die Sprache von einem großen Konzern entwickelt wurde, wurde sie zu einem Standard in der Welt

der Datenverarbeitung. Mit der Zeit wurden Verbesserungen vorgenommen, so daß die neuste Version FORTRAN 77 um einiges leistungsstärker geworden ist. FORTRAN ist trotz der Verbreitung von Personal-Comutern und trotz des Angebots dieser Sprache für PC von mehreren Softwarehäusern eine Sprache der großen Rechenanlagen geblieben.

Für den ATARI ST wird unter dem Namen ProFortran eine Version des FORTRAN 77 angeboten, die von der englischen Firma Prospero Software erstellt wurde. Der Sprachumfang von ProFortran entspricht dem Standard ANSIN-Norm Fortran 77.

Pascal

Ein Nachteil der vorhandenen BASIC- und Fortran-Versionen war es den Ablauf eines Programms nachzuvollziehen. Die häufigen Sprünge in Unterroutinen machten ein Programm sehr unlesbar. Heute gibt es BASIC-Versionen, die Prozeduren erlauben, und auch Fortran 77 kann besser strukturiert werden. Nach dieser Erkenntnis wurde Mitte der 60er Jahre die Sprache Algol geschaffen, der ein völlig neues Konzept zugrunde lag, dsa jedoch schnell Strukturen, Prozeduren, usw. implementierte. Man erkannte, daß auch diese Sprache an gewissen Stellen Schwächen hatte. Niklaus Wirth, der selbst an einer Algol-Version mitgearbeitet hatte, schuf 1970 an der Technischen Hochschule in Zürich PASCAL. Diese Sprache, benannt nach dem bekannten Philosophen und Mathematiker Blaise Pascal, wurde als Sprache für Informatiker, Mathematiker und Naturwissenschaftler konzipiert, ist aber beispielsweise ebenso für die Datenverwaltung geeignet. Ihr Hauptmerkmal liegt im übersichtlichen Aufbau, der durch spezielle Befehle ermöglicht wird. Dazu gehören verschiedene Anweisungen zur Schleifensteuerung sowie komfortable Verzweigungsmöglichkeiten. Dies bildet die Grundlage für eine strukturierte Programmierweise, der Pascal seine große Popularität verdankt. Ein anderer wesentlicher Pluspunkt von Pascal ist die Menge an Datentypen sowie die Struktur. Man kann Records definieren, die verschiedene Datentypen umfassen. Daraus ergibt sich eine Zusammensetzung von verschiedenen Datentypen sowie die Struktur. Man kann Records definieren, die verschiedene Datentypen umfassen. Daraus ergibt sich eine Zusammensetzung von verschiedenen Datentypen, die als selbst erfunden (je nach Zweck) betrachtet werden könnten.

Pascal erfreut sich einer großen Beliebthei bei ST-Anwendern und ist mehrmals vertreten.

Der erste Pascal-Compiler für den ST war GEMDOS-PASCAL von CCD. Damals gab es zu diesen Produkten keine Konkurrenz; insofern blieb dem Pascalprogrammierer keine Alternative als diese Version. Der Mangel, den diese Sprache damals hatte, wurde von der Softwarefirma CCD selbst nach und nach beseitigt. In der Anfangsphase leistete die Benutzeroberfläche EASY PAS, die von Jürgen Leonhard geschrieben wurde, große Hilfe. Dadurch ersparte man sich die mühselige Arbeit einer Batchprozedur. Die alte Pascal-Version wurde von einer völlig neuen und ausgereiften ersetzt. Diese Konkurrenzlosigkeit wurde später von METACOMCO erkannt. Das Softwarehaus brachte eine sehr umfangreiche Version dieser Sprache auf den Markt. Bei der CeBit 86 wurde von der in England ansässigen Firma PROSPERO eine andere Version, das in der IBM-Welt längst bekannte PROPASCAL angekündigt. Erst im September gelangte dieses Produkt auf den deutschen Markt. Das System war für Entwickler gedacht und ist schwieriger als die anderen Pascal-Versionen zu bedienen.

'C' ist nicht nur ein Buchstabe

Die Sprache 'C' wurde in den 70er Jahren von Dennis Ritchie bei Bell Laboratories, einer Tochtergesellschaft von AT&T, entwickelt. Man wollte das bis zu diesem Zeitpunkt in Assembler PDP11 geschriebene Betriebssystem UNIX umschreiben, um es auch für andere Maschinen lauffähig zu machen.

Bei der Entwicklung von 'C' wurde besonders darauf geachtet, eine Sprache zu schaffen, die so unabhängig wie möglich von jeder Hardware-Architektur bleibt. 'C' hat in fast allen Bereichen Anwendung gefunden. Von maschinennaher Programmierung, wo die schnelle Ausführung sehr wichtig ist, bis zu dem Einsatz in der Herstellung von CAD und Textverarbeitungsprogrammen findet man Programme, die in 'C' geschrieben wurden. Beim ATARI ST ist diese Sprache sehr häufig vertreten. Da die bequeme Bedienungsoberfläche des ST in 'C' geschrieben ist, eignet sich diese Sprache am besten für diesen Rechner. Eine ältere Version davon für die ST-Serie ist der Compiler von Digital Research. Er war Bestandteil des sogenannten Entwicklungspakets, das von ATARI selbst vertrieben wird. Einer der Nachteile dieses Compilers war seine konventionelle Umgebung: Wenn man einen Rechner besitzt, der eine sehr bedienungsfreundliche Oberfläche hat, sollte auch die Softdware darauf abgestimmt werden. Aus England kam später der GST C-Compiler, der das GEM vollständig nutzt, der aber keine Fließkomma-Arithmetik kannte. Die angepaßte Version der Lattice C fand auch bei den ATARI-Anwendern große Resonanz. Einer der beliebtesten 'C'-Compiler ist der sogenannte MEGAMAX C, der wegen seines ausführlichen Bibliotheksumfangs ein echtes Entwicklungssystem darstellt. Nicht nur die Bedienung ist bei dieser Version besser gestaltet, sondern auch die Compilier- und Linkzeiten sind viel kürzer als bei anderen Versionen. Ein Neuling für alle ST-verfügbaren C-Compiler ist das sogenannte Mark Williams C. Diese Version benutzt ein UNIX-ähnliches Shell, so daß jeder UNIX-Benutzer sehr schnell mit dieser Compiler-Version zurechtkommen wird. Die Programme, die mit Mark Williams C geschrieben werden, benötigen 2 bis 4 mal längere Zeit für die Vorgänge des Compilierens und Linkens. Möchte man Programme entwickeln, die einfach an MS-DOS-Betriebssysteme angepaßt werden müssen, so hat man mit Mark Williams ein umfangreiches und brauchbares Entwicklungssystem.

BASIC

BASIC wurde am Ende der 60er Jahre entwickelt. Der direkte Vorfahre ist eindeutig FORTRAN, obwohl die Sprache auch Elemente von ALGO besitzt. Kemeny und Kurtz, die BASIC am Dartmouth College in New Hampshire entwickelten, wußten damals wohl nicht, daß sie eine Sprache geschaffen hatte, die heute ein Standard für alle Home-Computer und die am meisten verbreitete Sprache im PC-Bereich ist. Die große Stärke liegt in erster Linie darin, daß es eine Interpreter-Sprache ist. Obwohl ihr Sprachumfang relativ groß ist, bleibt

BASIC sehr einfach zu erlernen. Der Vorteil eines Interpreters ist, daß ein Dialog Mensch-Rechner sofort stattfindet. Nach der Anweisung folgt eine sofortige Ausführung – ohne Zwischenphasen. Das alles hat diese Sprache vor allem bei Schülern und für Unterrichtszwecke sehr beliebt gemacht.

Ein großer Nachteil von BASIC war in der Anfangszeit, daß sehr viel Freiheit bei der Programmierung zugelassen wurde. Die Möglichkeit, in Unterroutinen und zurück zu springen, machte die Programme fast unlesbar. Solche Mankos wurden von den Entwicklern erkannt, so daß die neuen Generationen wesentlich erweitert wurden. Zur Zeit gibt es BASIC-Dialekte, die eher an Pascal erinnern.

Zum Lieferumfang des ST gehört ein BASIC-Interpreter, der von Digital Research für diesen Rechner-Typ erstellt wurde. Trotz seiner Bedienungsfreundlichkeit ist dieser Interpreter nicht brauchbar. Selbst ATARI hat die Probleme dieser BASIC-Version erkannt und hat angekündigt, daß in näherer Zukunft ein besserer und fehlerfreier Interpreter geliefert wird. Ein BASIC-Compiler wird von PHILON Inc., einer amerikanischen Softwarefirma, geliefert. Er ist recht umständlich zu bedienen und besitzt nur einen Vorteil: Der Sprachumfang dieser Version entspricht dem des bekannten MBASIC von Microsoft.

Für den ST-Besitzer hat der GfA-BASIC-Interpreter wegen seiner Leistungsmerkmale eine Sonderstellung erreicht. Diese BASIC-Version benutzt keine Zeilennummern mehr, so daß bei Sprüngen auf Labels verwiesen werden muß. Es wird nur ein Befehl pro Zeile akzeptiert, was der besseren Lesbarkeit eines Programms dient. Man kann in GfA-BASIC mit echten Prozeduren programmieren, die Parameter akzeptieren und einen oder mehrere Rückwerte zurück geben. Mit dem neuen Compiler für diesen Interpret er hat sich für BASIC eine völlig neue Perspektive eröffnet. Zur Zeit ist auch dieser Compiler leider nicht absolut fehlerfrei.

Die jüngste deutsche Entwicklung ist OMIKRON-BASIC. Dieser Interpreter wird als ROM-Modul geliefert, so daß er beim Einschalten des Rechners immer sofort zur Verfügung steht. Das Omikron-BASIC sticht vor allem durch seine hervorragenden mathematischen Fähigkeiten hervor. Eine umfangreiche Bibliothek ist im Lieferumfang dabei. Dazu gehört ein Modul zur Erstellung von index-sequentiellen Datei (ISAM).

Modula-2

Modula ist eine Weiterentwicklung von Pascal. Wie bei den eben genannten Sprachen ist auch hier N. Wirth der geistige Vater. 1978 wurde Modula der Öffentlichkeit in Form einer Maschine gezeigt, die nur diese Sprache beherrschte. Modula ist stark nach Pascal aufgebaut, aber in der Leistung wesentlich verbessert. Die Programme werden in kleine Einheiten zerlegt, die man Modula nennt. Jedes Modul ist völlig unabhängig und kann getrennt übersetzt werden. Die Module entsprechen etwa Bibliotheksstücken, die jederzeit in ein anderes Programm implementiert werden können. Die Prozeduren können bei Modula als Datentypen deklariert werden, so daß eine sehr große Flexibilität angeboten wird. Durch die implementierten Coroutinen können in Modula nebenläufige Prozesse (Multitasking) realisiert werden.

TDI bietet für alle ST-Rechner ein Modula-2 System und zusätzliches Toolkit an. Damit steht dem ST-Anwender eine der modernsten Sprachen zur Verfügung. Die GEM-Oberfläche wird bei Modula-2 von TDI vollständig unterstützt.

FORTH, die vierte Dimension

Charles Moore schuf diese Sprache in den 60er Jahren als Konkurrenz zu den damals vorhanden Sprachen, die für Steuerungszwecke ineffektiv waren. Der erste Einsatz dieser „vierten Generation" (daher der Name) war die Ansteuerung eines Teleskops in einem Observatorium. Wie die anderen höheren Programmiersprachen besitzt FORTH Sprachelemente bzw. feste Vereinbarungen, die der Anwender beachten muß. Ein wesentlicher Punkt, in dem die Sprache sich von anderen unterscheidet, ist, daß sich aus den definierten Sprachelementen neue Funktionen ableiten lassen. Dadurch hat der Programmierer eine unbegrenzte Möglichkeit, verschiedene Funktionen selbst zu definieren. Ein anderer Vorzug von FORTH liegt in der Maschinennähe, die es erlaubt, die jeweilige Prozessorarchitektur effektiv auszunutzen, und in der daraus resultierenden hohen Verarbeitungsgeschwindigkeit.

Für den ST werden über ein halbes Dutzend verschiedener FORTH-Versionen angeboten. 4★FORTH von der Dragon-Group gehört zu den ersten Sprachen, die für den ST erhältlich waren. Diese Version von FORTH stellt sich als kompaktes Programmentwicklungssystem dar, das die Möglichkeiten des ST effektiv nutzt. Von Data Becker wird FORTH ST angeboten, eine sehr leistungsfähige Version, die unter GEM läuft.

Aus unserem Public-Domain Service ist ebenfalls eine Version von FORTH-83 zu erhalten.

APL: Eine Einführung in die Hieroglyphen

APL steht für „A Programming Language", was sich natürlich von sämtlichen Interpretern und Compilern behaupten läßt. APL wurde ursprünglich als universelle Notation zur Vermittlung von logischen (mathematischen) Zusammenhängen entwickelt und erstmals am legendären IBM 360 implementiert. Seither hat sich die Sprache auf Computern dieser Größenordnung rapide verbreitet. Dennoch blieb APL großen Anwenderkreisen unbekannt. Das Erscheinen sogenannter „Supermicros" Ende der 70er Jahre bescherte APL eine Verbreitung in der dezentralen Datenverarbeitung.

Die Interpreter-Sprache APL ist ständig verbessert worden. Bei APL-Befehlen handelt es sich nicht um pseudo-englische Sprachschöpfungen wie „ON... GOSUB", sondern um Symbole: Eine Bildersprache also, wie sie im alten Ägypten verwendet wurde. Die Programme in APL werden modular aufgebaut. Die einzelnen Routinen werden als Funktionen berechnet – und es sind welche, da sie im Programm mit exakt derselben Syntax verwendet werden wie die im Sprachumfang enthaltenen Funktionen. Dieser Umstand deutet schon an, daß (ähnlich wie in FORTH) APL vom Anwender erweitert und ausgebaut wird.

APL-Programme sind meist sehr kurz; üblicherweise nimmt ein vergleichbares Programm in BASIC oder Pascal etwa zehnmal soviele Zeilen in Anspruch.

Für den ST wird von GDAT (Gesellschaft für dezentrale Daten/Technik) eine Version dieser Sprache angeboten. APL/68000 beruht auf dem IBM-

Standard des APL.SV. Er ist vollständig implementiert. Diese Version besitzt ein eigenes Dateisystem. Das APL/68000-Dateisystem arbeitet indexsequentiell und bietet sämtliche denkbaren Kontrollfunktionen, die für große Datenbanken vonnöten sein könnten.

LOGO: Nicht nur für Kinder

LOGO wurde Ende der 60er Jahre von Seymour Papert, Professor im Institut of Technology, Massachusetts, entwickelt, einem Labor, das sich mit künstlicher Intelligenz beschäftigt. LOGO gehört zu den Interpreter-Sprachen und ist daher sehr schnell zu beherrschen.

Papert, Mathematikprofessor und ehemaliger Student des schweizerischen Psychologen Jean Piaget (bekannt als Vater der modernen Psychologie), strebte nicht nach einer neuen Sprache, die das Optimum eines Computers bilden sollte, sondern eher nach einem pädagogischen Werkzeüg, das die intellektuellen Fähigkeiten der Studierenden entwickeln sollte – genauso, wie der tatsächliche Lernprozeß komplizierte Vorgänge erleichtert.

So entstand eine Programmiersprache, die im Dialog-Betrieb mit dem Anwender arbeitet und in der Lage ist, schwierige Prozesse anschaulich zu lösen. Dazu dienen kurze Befehle in Form von einfachen und bekannten Begriffen. Beim damaligen Stand der Technik war es unmöglich, solche Lösungen auf einem Sichtgerät darzustellen. Papert schuf eine mechanische „Schildkröte", an deren Unterseite ein Stift befestigt war. Beim Fahren auf einer Papierunterlage hinterließ sie Linien. Heute hat sich diese motorisierte „Schildkröte" in ein stilisiertes Dreieck verwandelt, das auf dem Bildschirm Figuren zeichnen kann. Zusätzlich zu den Befehlselementen, die zur Verfügung stehen, kann man die Sprache durch selbst definierte Funktionen erweitern, die in jedes Programm eingebaut werden können. LOGO hat heute im Bereich der künstlichen Intelligenz seinen Platz als Untermenge der Sprache Prolog.

ATARI selbst vertrieb für seine ST-Computer-Reihe eine LOGO-Version, die von der Software Firma Digital Research entwickelt wurde. Dr. LOGO ist GEM-implementiert und einfach zu erlernen. Pull-Down-Menüs sowie die Bedienung durch die Maus vereinfachen, vor allem für den Neuling, den ersten Einstieg. Ein großer Nachteil dieser Version ist die geringe Ausführungsgeschwindigkeit.

LISP und PROLOG: Sprachen der Künstlichen Intelligenz

LISP und PROLOG sind ohne Zweifel die Muttersprachen der künstlichen Intelligenz. Am Anfang hatten sie nicht dieselbe Verbreitung gefunden wie andere Sprachen. Seit einigen Jahren hat sich der Computer-Fachmann immer mehr mit dem Thema Computer-Intelligenz beschäftigt und die Sprachen wurden neu entdeckt. Sie gehören zu sogenannten applikativen bzw. deklarativen Sprachen. Bei den imperativen Sprachen (FORTRAN, BASIC, PASCAL, FORTH, C, ...) wird ein Problem nach einem bestimmten, vom Programmierer festgelegten Ablauf gelöst. Es hängt von der Fähigkeit des Programmierers ab, die verschiedenen Ausnahmesituationen, an die ein Programm während seines Ablaufs stoßen könnte, zu berücksichtigen. Hat der Programmierer darin wenig Erfahrung, wird sich das Programm entsprechend wenig zuverlässig verhalten.

Ein Programm, das in einer aplikativen Programmiersprache geschrieben wurde, besitzt keinen festen Algorithmus, sondern eher eine Softwarewelt aus Regeln und Schlußfolgerungen. Die eigentliche Lösung von Fragestellungen besorgt dann ein (in den Compiler oder Interpreter bereits integrierter) backtracking Algorithmus.

LISP ist die älteste KI-Sprache (und nach FORTRAN die zweitälteste Computersprache überhaupt) und existiert in vielen Versionen, in denen meist auch reichlich imperative Sprachelemente vorhanden sind. LISP entstand zunächst aus einer Kombination von FORTRAN und IPL. John McCarthy, der Vater von LISt Processor (LISP), vereinigte eine der wichtigsten Eigenschaften von IPL, die 'Listen', als Datenstrukturen mit den Grundfunktionen von FORTRAN. 1960 war ein komplettes LISP-System fertig, das aus einem Interpreter und Compiler bestand.

Auch in diesem Bereich werden für den ST verschiedene Versionen angeboten. Eine der älteren, die für diesen Rechner existieren, ist zum Nulltarif zu erhalten. Es handelt sich um XLISP, die sich auf Diskette Nr. 7 unserer Public-Domain befindet. XLISP wurde in C geschrieben und ist leicht an andere Rechner anzupassen. Die Dokumentation wird auf der Diskette in Form eines Files mitgeliefert und umfaßt circa 43 Seiten.
Die Softwarefirma Metacomco bietet unter dem Namen Cambridge LISP eine Version, die ursprünglich für Großrechneranlagen entwickelt wurde. Es handelt sich im wesentlichen um eine Erweiterung von Standard-Lisp. Cambridge LISP ist vollständig menügesteuert und wird mit einem in Englisch geschriebenen Handbuch geliefert.
Ein weiteres Angebot kommt aus Deutschland: LISPAS II. Tommy-Software, ein in der Mainmetropole ansässiges Softwarehaus, bietet für den ST eine LISP-Version an, die durch ihre Implementation von GEM einen einfachen Einstieg in das Neuland der künstlichen Intelligenz ermöglicht.

PROLOG steht für 'Programming in Logic' und bezieht sich auf eine Idee, die Anfang der 70er Jahre in den Kreisen der Programmierer ziemlich neu war: Das Programmieren nach einer 'logischen' Denkweise. PROLOG wurde jedoch wiederentdeckt, als die Japaner diese Sprache als Sprache der 'fünften Generation' und als Bestandteil ihrer neuen Technologie gewählt hatten. PROLOG befreit den Programmierer vom imperativen Ballast seines Problems bzw. seines Problem-Lösungswegs. Man könnte sogar sagen, daß mit PROLOG der Traum erreicht worden ist, einen Computer programmieren zu können, ohne eine bestimmte Programmiersprache lernen zu müssen: Ein völlig neues Konzept.
Auch eine PROLOG-Version ist von unserem PD-Service zu erhalten. TOY-Prolog ist eine Implementation des bekannten DEC-10 Dialekts. Diese Version ist nicht sehr schnell, doch trotzdem für den PROLOG-Neuling ein gutes Werkzeug zum Einsteigen in die Welt der fünften Generation.

Mit der PROLOG-Implementierung „MProlog Logic-Lab" steht auf dem ATARI ST ein Labor für künstliche Intelligenz zur Entwicklung von Expertensystemen zur Verfügung. MPROLOG ist syntaktisch kompatibel zum Prologstandard, der im Handbuch „Programming in Prolog" von Clocksin und Mellisch beschrieben wurde. Mit einer Ausführungsgeschwindigkeit von ca. 800 LIPS (Logical Infere-

ces per Second) kann die Interpreterversion mit mancher Prolog-Implementierung auf Minicomputern Schritt halten.

Ein Rückblick in die Vergangenheit

Wie schon oben erwähnt wurde, war Assembler der erste Schritt zu einer vernünftigen Kommunikation mit dem Rechner. Assembler hat den Programmierer von der Codierung der Befehle befreit, und die Programme um einiges deutlicher und verständlicher gemacht. Die Programme waren besonders schnell in ihrer Ausführung, und an die Maschine sehr gut angepaßt. Der zuerst genannte Punkt ist zweifellos von Vorteil, aber der maschinenabhängige Code macht diese Programme nicht übertragbar. Da jeder Prozessortyp seine eigenen Befehle und Adressierungsarten hat, wurde somit vom Hersteller eine eigene Assemblersprache entwickelt. Der Assemblerprogrammierer muß deshalb, wenn er auf verschiedenen Systemen mit unterschiedlichen Prozessoren arbeitet, mehrere Assemblersprachen lernen. Trotz aller Nachteile bleibt Assembler die effektivste Art einen Rechner zu programmieren. Da der Assembler direkt Maschinencode erzeugt, sind Assemblerprogramme die schnellsten und kürzesten. Assembler wird in vielen Anwendungen eingesetzt, die zeitkritisch oder von einer Hochsprache aus nicht erreichbar oder lösbar sind.

Im sogenannten Entwicklungssystem, das von ATARI vertrieben wird, ist ein Assembler (CP/M 68K Assembler) der Software Firma Digital Research enthalten. Zum Lieferumfang des Assemblers gehört ein Editor und Linker. Der Assembler beherrscht den Motorola Standard-Sprachumfang und stellt damit ein sehr brauchbares Werkzeug zur Herstellung von Programmen in Assembler dar.

Von der Firma OMIKRON wird unter dem Namen IDEAL ein Assembler-Paket angeboten. Es handelt sich um einen integrierten Debugger, Editor, Assembler und Linker. Dem im IDEAL integrierten Debugger kann man als einen der besten auf dem Markt bezeichnen. Der Editor ist, wie das ganze IDEAL-Paket, im schlichten TOS-Design gehalten, was aber nicht heißt, daß es ihm an Leistung mangelt. GST, ein Softwarehaus, das in England ansässig ist, und eine Fülle von Produkten für den ATARI anbietet, hat für den Assembler Programmierer ein Makro-Assembler-System, das äußerst professionell ist. Auf der Programmdiskette befinden sich der Assembler, der Editor, der Linker, eine Makrobibliothek und das Shell-Programm. Der Editor scheint ein Vorläufer von 1st_ Word zu sein und ist genau so leicht zu bedienen. Beim GST-Assembler sind die Möglichkeiten gegeben, mit Makros, vielen Pseudo-Opcodes und Funktionen, ähnlich wie bei einem Compiler, zu programmieren. Der Linker bietet eine Vielzahl von Funktionen, um Programme zu verbinden, Bibliotheken anzulegen und Programmteile aus Bibliotheken einzubinden.

Der Metacomco Macro Assembler bietet dem Assembler-Programmierer hervorragende Möglichkeiten. Der Assembler unterstützt zahlreiche Pseudo-Opcodes und lokale Labels. Dadurch wird die bedingte Assemblierung erst ermöglicht. Als Linker findet man den gleichen, der im GST-Assembler-Paket enthalten ist, wieder. Dieser besticht durch die Vielzahl seiner Optionen.

(MM)

Sprache	DM	Bezugsquelle
Assembler:		
DR. Assembler (Ass 68) (★)	871,–	diverse Händler
GST Assembler	149,–	diverse Händler
SEKA Assembler	148,–	diverse Händler
Profimat	99,–	Data Becker
IDEAL Assembler	99,–	OMIKRON
MCC Assembler	168,–	diverse Händler
FORTRAN:		
Fortran 77	548,–	Softline
LISP:		
Cambridge LIPS	498,–	Softline
LISPas II	298,–	Tommy Software
XLISP	P. D.	ST Computer
Forth:		
4★Forth Level 1	348,–	diverse Händler
4★Forth Level 2	548,–	diverse Händler
Volks-Forth	P. D.	ST Computer
Forth ST	99,–	diverse Händler
BASIC:		
ST BASIC (★★)	–	ATARI
GFA BASIC	196,–	GFA Syst.
Omikron BASIC	229,–	OMIKRON
Philon BASIC	348,–	diverse Händler
C:		
DR. C Compiler (★)	871,–	diverse Händler
MEGAMAX C	495,–	diverse Händler
Mark Williams C	398,–	Softline
Lattice C	348,–	diverse Händler
PASCAL:		
ProPascal	448,–	diverse Händler
Pascal +	249,–	ATARI
Pascal	149,–	ATARI
MCC Pascal	198,–	diverse Händler
UCSD Pascal	349,–	diverse Händler
Modula:		
TDI Modula	298,–	diverse Händler
Modula II Toolkit	148,–	diverse Händler
Prolog:		
M Prolog	998,–	Epsilon
TOY Prolog	P. D.	ST Computer
Logo:		
DR. Logo	99,–	diverse Händler
APL:		
APL 68000	ca. 1000,–	GDAT

Für die Richtigkeit der Preise können wir keine Gewähr leisten

★ Ist Bestandteil des sogenannten Entwicklungspakets
★★ Wird beim Kauf des STs mitgeliefert
P. D. = Public Domain Programm

Klein, aber fein:

Der ST-Oszillograph und Sound-Sampler



Die Einsatzgebiete eines Computers werden immer vielfältiger. Recht neu ist die Verwendung als Meßgerät oder digitales Tonbandgerät. Wir testeten ein neues Zusatzmodul, das den ATARI ST nicht nur in ein Speicheroszilloskop verwandelt...

Das Konzept

Die Firma „Microcomputer-Labor" aus Saarbrücken hat einen Meßvorsatz zum ST entwickelt, der analoge Signale in digitale Werte umwandelt und diese über den Parallelport dem Rechne rzur Verfügung stellt. Die mitgelieferte Software erlaubt es nun, die von der Hardware gelieferten Werte zeitabhängig auf dem Bildschirm darzustellen. Die Höhe der an der Meßspitze anliegenden Spannung wird also in ihrem zeitlichen Verlauf dargestellt. Dabei erlaubt die Softwaer ein komfortables Einstellen sowohl der Zeitachse (Horizontale) als auch der Spannungsachse (Empfindlichkeit der Meßspitze, Vertikale). Diese beiden Achsen sind kalibriert, d. h. durch ein einblendbares Raster können genaue Spannungs- bzw. Zeit-Werte vom Bildschirm abgelesen werden. So wird der ST zum **Oszilloskop!** Da man auch extrem niedrige Frequenzen im Subsonic Bereich oder kurzzeitige aperiodische Vorgänge im RAM-Speicher des ST speichern und anschließend in Ruhe betrachten und auswerten kann, sind die Voraussetzungen eines **Speicheroszilloskops** ebenfalls erfüllt. Doch der ST-Oszillograph kann noch mehr. Hatte man schon die Möglichkeit geschaffen, beliebige Spannungswerte digital abzuspeichern, so fehlte nur noch die entsprechende Software, um sie als Tonfolge über den Monitorlautsprecher wiederzugeben. Auch diese Option wird mitgeliefert, so daß man neben dem ST-Oszillograph auch noch einen ST-**Sound-Sampler** erhält.

Bedienung und Anschluß der Hardware

Der Meßvorsatz besteht aus einem kleinen Plastikgehäuse mit integrierte Meßspitze. Hier ist auch die gesamte Elektronik untergebracht. Ein neunstufiger Schalter dient zur Einstellung der Eingangsempfindlichkeit zwischen 0,1 und 50 Volt pro Einheit, ein weiterer Regler dient zum Festlegen der Nullinie (Y-Position). Mit einem sogenannten AC/DC-Schalt er läßt sich ein eventuell vorhandener Gleichspannungsanteil ausfiltern. Ferner ist noch eine Leuchtdiode als Betriebskontrolle und eine Chinch-Buchse als NF-Eingang für den Sound-Sampler vorhanden. Der Meßvorsatz wird über ein 1,8 Meter langes Kabel am Parallelport angeschlossen. Seine Betriebsspannung erhält das Gerät über den Joystick-Port.

Der ST als Oszilloskop

Bei der Verwendung des Meßvorsatzes als (Speicher-) Oszilloskop, was wohl das Haupt-Einsatzgebiet sein dürfte, stehen zwei Betriebsarten zur Verfügung (siehe Bild 1). Im **Direkt**-Modus werden alle Signale, die an der Meßspitze anliegen, sofort angezeigt. Auf Tastendruck kann diese Betriebsart verlassen werden, um z. B. die Abtastfrequenz zu verändern oder auch den **Store**-Modus (engl. store = speichern) mit der Maus anzuwählen. In dieser Betriebsart, quasi als digitales Speicheroszilloskop, werden 50 Bildschirmseiten aufgenommen und gespeichert. Anschließend können diese Seiten mittels der Maus horizontal „durchgescrollt" und dargestellt werden. Da das Aufnehmen der Messwerte im Extremfall fast drei Tage dauern kann (Timebase = 500 s) ist ein Stoppen durch Drücken der Leertaste möglich. Eine einmal aufgenommene „Kurve" kann auf Diskette abgespeichert und später wieder von ihr geladen werden.

Grenzwerte

Um die Qualität eines Oszilloskops zu bewerten, werden vor allem zwei wichtige Kriterien herangezogen: Zum einen die maximale Empfindlichkeit des Eingangsverstärkers, zum anderen die maximale Abtastfrequenz der Zeitablenkung (Timebase), in der sich die Grenzfrequenz des zu messenden Signals widerspiegelt. Bei diesen Kriterien schneidet der ST-Oszillograph nicht gerade gut ab, man könnte ihn mit ausreichend bewerten. Die Eingangsempfindlichkeit von 0,1 Volt pro Einheit (Linie) ist auch in der Niederfrequenztechnik (NF) nicht immer

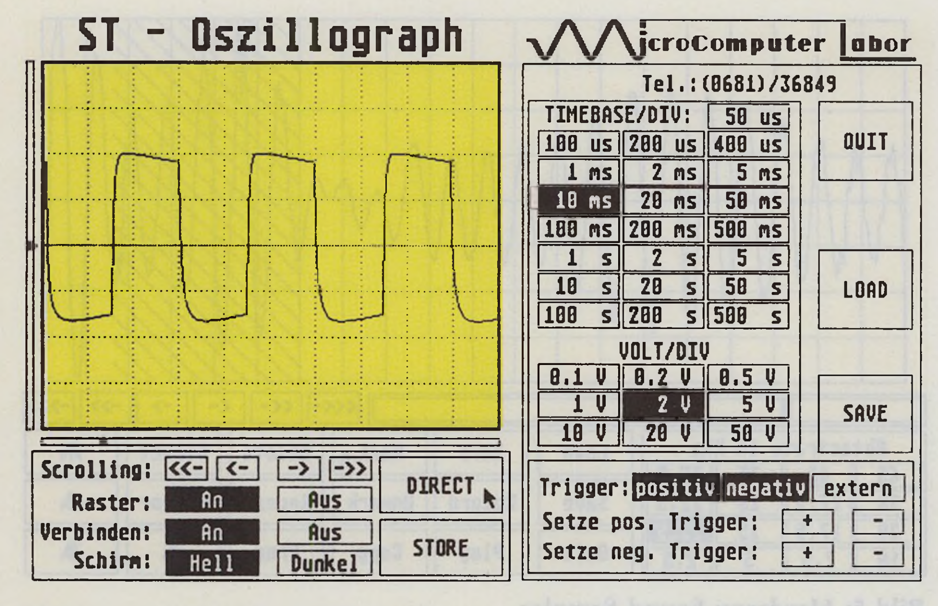


Bild 1: Hardcopy ST Oszillograph

Das Triggern

Unter Triggern versteht man die Fähigkeit, ein stehendes Bild zu produzieren bzw. ab einer bestimmten Signalspannung mit der Messung zu beginnen. Die Software ermöglicht im Direkt-Modus eine automatische Triggerung. Bei Betrieb als Speicheroszilloskop kann der Triggerpegel, sowohl für den negativen- als auch für den positiven Spannungsbereich, getrennt vorgegeben werden. Außerdem ist es in dieser Betriebsart möglich, einen Triggerimpuls von außen zuzuführen, indem der Pin 22 der seriellen Schnittstelle von „High" nach „Low" geschaltet wird. Es sind somit alle Triggermöglichkeiten vorhanden, die auch sehr gut arbeiten.

ausreichend. Die Software erreicht maximal 60 000 Messungen pro Sekunde (Timebase = 50 Mikrosekunden pro Einheit), womit **theoretisch** die Grenzfrequenz des Eingangssignals bei 30 KHz liegt. Wir haben uns, um sicher zu gehen, eine Dreiecksspannung gleichzeitig mit einem herkömmlichen Oszilloskop (HM 512) und dem ST-Oszillograph angesehen. Dabei ergab sich bis zu einer Frequenz von etwa 8000 Hz eine gute Übereinstimmung, sowohl im Kurvenverlauf als auch beim Ablesen der Spannungshöhe bzw. der Frequenz (siehe Abbildungen 2, 3 und 4). Zu bemerken ist dabei, daß die Zeitablenkung teilweise nicht genau ist. So läßt sich z. B. die Frequenz einer 8000 Hz-Dreieckschwingung bei Ablenkung von 100 Mikrosekunden pro Einheit genau ermitteln, bei einer Ablenkung von 200 Mikrosekunden dagegen errechnet sich eine um ca. 20 Prozent abweichende Frequenz. Durch die eingeschränkte Bandbreite läßt sich der Meßvorsatz als Ersatz eines normalen Oszilloskops nur sehr bedingt empfehlen. Die Stärken des ST-Oszillographen liegen jedoch in der Betriebsart als Speicheroszilloskop, vor allem, wenn man bedenkt, daß solche Geräte das 10- bis 20fache kosten.

Ein Speicheroszilloskop ist in der Lage, unperiodische und/oder extrem langsame Spannungsänderungen wie z. B. die Entladekurve eines Akkus oder das Einschaltverhalten von Netzteilen aufzunehmen und sichtbar zu machen. So beträgt die maximale Meßdauer 69,5 Stunden. Das sind fast drei Tage! Somit eignet sich der ST-Oszillograph gut für Langzeitbeobachtungen. Natürlich wird die Abtastfrequenz bei einer solchen Langzeitmessung über 69,5 Stunden entsprechend gering, d. h. in diesem Fall z. B. wird alle 15,6 Sekunden ein Meßwert aufgenommen.

Ausdrucken einer Kurve

Mit dem ST Oszillo**graph** hat man die Möglichkeit, eine erfaßte Kurve graphisch, also schwarz auf weiß auf Papier auszugeben. Allerdings ist dies nur als Hardcopy durch Drücken von Alternate und Help möglich. Zuvor muß natürlich der Meßvorsatz abgezogen und ein Drucker angeschlossen werden. Das Drucken einer Kurve ist folglich etwas umständlich und hätte eventuell durch einen entsprechenden Adapteranschluß und eine geeignete Druckerroutine besser gelöst werden können.

Übrigens...

Es wird ein GFA-BASIC-Programm mitgeliefert, um den ST-Oszillographen in eigene Programme einzubinden. Die gemessenen Daten können dann in einer sequentiellen Datei auf Diskette gespeichert und ausgewertet werden.

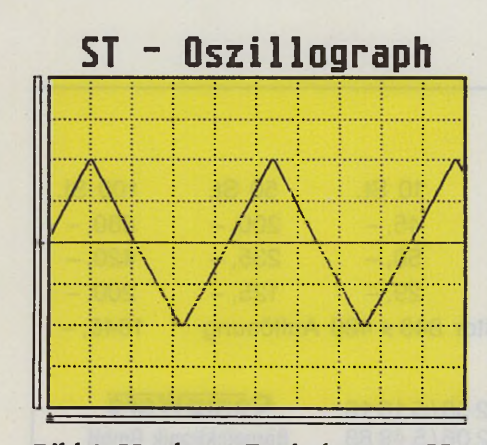


Bild 2: Hardcopy Dreiecksp. 250 Hz

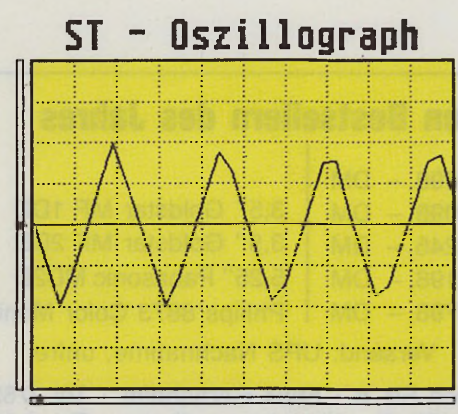


Bild 3: Hardcopy Dreiecksp. 8000 Hz

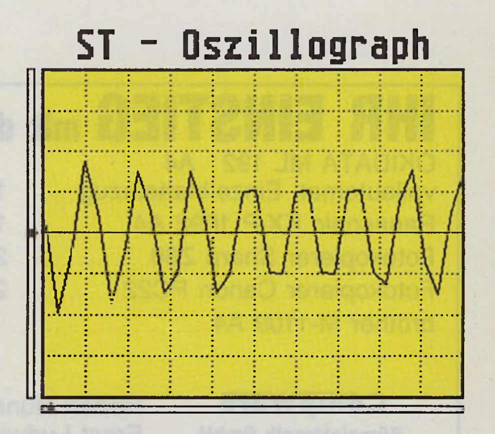


Bild 4: Hardcopy Dreiecksp. 16000 Hz

Der Sound-Sampler

Mit dem Sound Sampler können Töne oder Geräusche, kurz gesagt jegliche Art von NF-Signalen, digitalisiert bzw. aufgenommen und über den Monitorlautsprecher wiedergegeben werden. Das „Arbeitsfeld" des Sound Samplers ist auf Bild 5 zu sehen. Die oberen zwei Dreitel des Bildschirms dienen zum Aussteuern des Eingangssignals und zum späteren „Bearbeiten" des Stückes. Die Abtastfrequenz (sie gibt die Häufigkeit der Messungen an) ist neben der Auflösung des verwendeten A/D-Wandlers ausschlaggebend für die Tonqualität. Sie läßt sich in sechzehn Stufen zwischen 2,5 und 45 KHz einstellen. Bei einem freien Speicherplatz von 826 KByte (520 ST+ oder 1040 ST/F) lassen sich bei einer Abtastrate von 45 KHz ca. 19 Sekunden aufnehmen. Bei 2,5 KHz steigt die Zeit auf gute 5,5 Minuten, jedoch ist die Klangqualität dann zwangsläufig sehr schlecht. Da ein 8-Bit-A/D-Wandler verwendet wird, ist auch bei einer hohen Abtastfrequenz niemals HiFi-Qualität zu erlangen. Dennoch eignet sich der Sound-Sampler gut für Demonstrationszwecke oder zum Einsteigen in diese moderne Art der Musikkonservierung. Übliche Funktionen wie Laden und Speichern auf Diskette oder Kopieren, Spielen und Löschen vorher markierter Bereiche ist durch die in GEM eingebundene Software problemlos möglich.

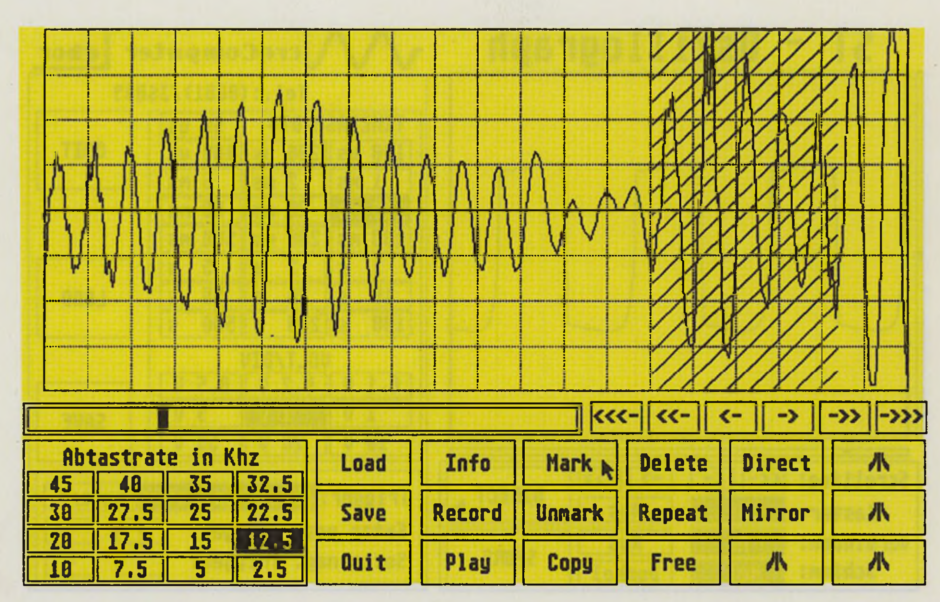


Bild 5: Hardcopy Sound Sampler

Fazit

Ein Oszilloskop ist wohl das wichtigste und universellste Meßgerät einer jeden Elektronik-Werkstatt. Leider sind solche Geräte auch nicht ganz preiswert. Daher kann dem Hobby-Bastler diser recht preisgünstige Meßzusatz bei bestimmten niederfrequenten Problemen hilfreich sein. Vor allem in der Betriebsart als Speicheroszilloskop lohnt sich der ST-Oszillograph. Und immerhin erhält man einen Sound Sampler gratis!

Die technischen Daten des ST-Oszillographen

Bildschirmdarstellung:
8 x 10 Linien, 50 Seiten Bildschirmspeicher

Y-Verstärker:
0,1 bis 50 Volt pro Einheit, kalibrierbar

Timebase:
50 Mikrosekunden pro Einheit bis 500 Sekunden pro Einheit

Trigger:
intern (automatisch), manuell und extern

Meßgeschwindigkeit:
maximal 60 000 Messungen pro Sekunde

Meßdauer:
1 Millisekunde bis 69,5 Stunden

Auflösung:
8 Bit (256 Steps)

Software:
Oszillograph: nur Monochrom-Monitor
Sound Sampler: Monochrom- und Farbmonitor

Preis:
DM 448,–

Hersteller:
Microcomputer-Labor
Schumannstraße 23
D-6600 Saarbrücken

Vertrieb:
diverse Händler

(UB)

Gemkurs

Dieses Mal werde ich Ihnen, wie versprochen, einige Spezialitäten präsentieren, die der Umgang mit GEM-Objektstrukturen ermöglicht. Voraussetzung ist, daß sie die Grundlagen im vorangegangenen Teil verdaut haben.

Ich habe wieder ein kleines Beispielprogramm geschrieben, das die erweiterten Möglichkeiten der Objektbeeinflußung demonstriert. Da das Listing nicht allzu lang ist, konnte es komplett abgedruckt werden (Listing 2). Benötigt wird wiederum das (hier nicht abgedruckte) Modul **inigem.c** aus Teil 1 des Kurses. In Listing 1 finden sie zur besseren Orientierung das Include File mit den Namen der Objekte und Objektbäume.

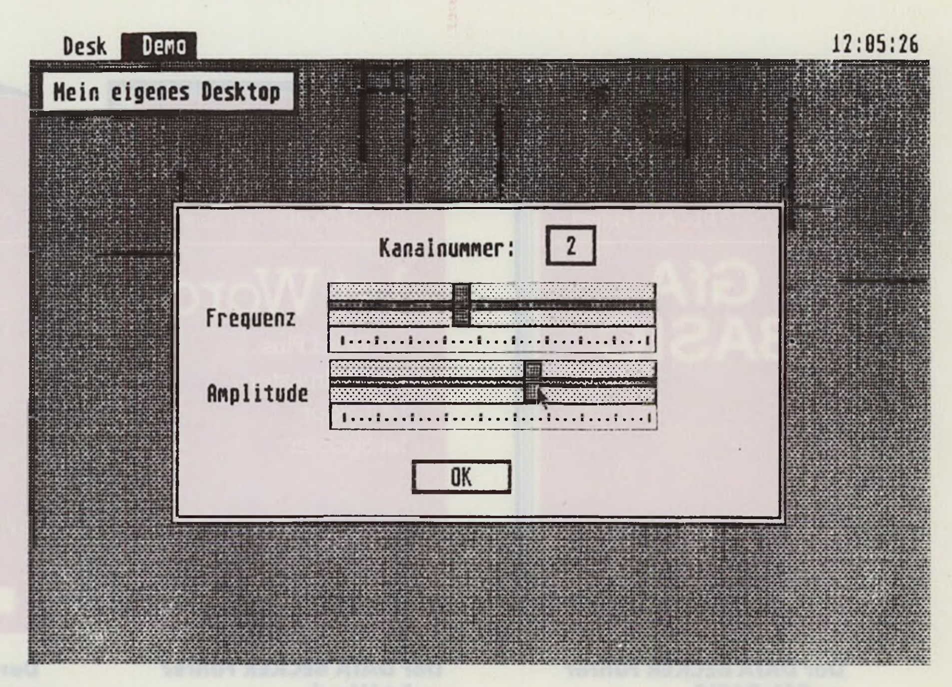


Bild 1

Das TOUCHEXIT Flag

Wer sich die möglichen Objektflags in der GEM-Dokumentation genauer angesehen hat, dem ist sicher ein Flag namens **TOUCHEXIT** aufgefallen. Was hat es damit auf sich? Ein normaler Exit-Button wird erst dann wirksam und der **form_do()** Aufruf wird erst dann verlassen, wenn man den Mausknopf drückt und dann wieder losläßt. Ein mit TOUCHEXIT gekennzeichnetes Objekt ermöglicht das Verlassen von **form_do()** bereits dann wenn der Mausknopf nur gedrückt wird, also vor dem Loslassen des Knopfes. Wenn also der Benutzer Ihres Programms den Mausknopf über einem TOUCHEXIT-Objekt drückt erhalten Sie sofort die Kontrolle über das weitere Geschehen zurück. Meist wird man das programmtechnisch derart gestalten, daß man um den **form_do()** Aufruf eine Schleife programmiert und innerhalb der Schleife die TOUCHEXIT Behandlung vornimmt. Diese Schleife läuft solange, bis der Benutzer ein echtes EXIT-Objekt anklickt.

```
 1 : #define SMILER      0                /* Baumnr.  0  =  Dialog     */
 2 : #define           SMILEY1      2     /* Objekt      =  BOX        */
 3 : #define           SMILEY2      3     /* Objekt      =  BOX        */
 4 : #define           SMILEY3      4     /* Objekt      =  BOX        */
 5 : #define           SMILEY4      5     /* Objekt      =  BOX        */
 6 :
 7 : #define MENU        1                /* Baumnr.  1  =  Menue      */
 8 : #define           MDESK        3     /* Objekt      =  TITLE      */
 9 : #define           MDEMO        4     /* Objekt      =  TITLE      */
10 : #define           MINFO        7     /* Objekt      =  STRING     */
11 : #define           MSOUND      16     /* Objekt      =  STRING     */
12 : #define           MSMILE      17     /* Objekt      =  STRING     */
13 : #define           MQUIT       19     /* Objekt      =  STRING     */
14 :
15 : #define INFO        2                /* Baumnr.  2  =  Dialog     */
16 :
17 : #define SOUND       3                /* Baumnr.  3  =  Free       */
18 : #define           FEXT         5     /* Objekt      =  BOX        */
19 : #define           FBAR         6     /* Objekt      =  BOX        */
20 : #define           FSLIDE       7     /* Objekt      =  BOX        */
21 : #define           AEXT         9     /* Objekt      =  BOX        */
22 : #define           ABAR        10     /* Objekt      =  BOX        */
23 : #define           KANAL       12     /* Objekt      =  BOXCHAR    */
24 : #define           ASLIDE      15     /* Objekt      =  BOX        */
25 :
26 : #define MYDESK      4                /* Baumnr.  4  =  Free       */
```

Listing 1

In den Zeilen 231-274 von Listing 2 können Sie sehen, wie man das aus den letzten beiden Teilen bekannte **hndl__dialog()** auf diese Art leicht erweitern kann. Nach der Rückkehr aus **form__do()** wird festgestellt ob das EXIT-Objekt das TOUCHEXIT-Flag gesetzt hat. In diesem Fall wird eine Funktion **hndl__touchexit()** aufgerufen. Sie können die Funktion **hndl__dialog()** ohne weiters in Ihren eigenen Anwendungen benutzen, Sie müssen nur jeweils **hndl__touchexit()** entsprechend anpassen.

Einen Pferdefuß hat die Sache noch: GEM prüft bei TOUCHEXIT-Objekten, ob die Maustaste sofort losgelassen und erneut gedrückt wird, ob also ein „Doppelklick" vorliegt. Wenn ja, wird bei der von **form__do()** zurückgelieferten Objektnummer das höchste Bit (Bit 15) gesetzt. Falls Sie dieses Feature nicht verwenden, sollten Sie Bit 15 unbedingt wegmaskieren (& 0x 7FFF), sonst gibt es unangenehme Überraschungen. Eine Anwendung des Doppelklicks sehen sie beim GEM-Fileselektor. Wenn ein Filename doppelt angeklickt wird, wird der Fileselektor sofort verlassen und der entsprechende Name zurückgeliefert. Auf diese Weise spart der Benutzer das **OK**.

Noch eine rein technische Bemerkung: Wie schaltet man das TOUCHEXIT-Flag überhaupt ein? Möglichkeit 1: Man kann es direkt im Resource Construction Set (RCS) setzen. Möglichkeit 2: Man macht es vom Programm aus mit:

tree[item].ob__flags |= TOUCHEXIT;

Bei Verwendung von TOUCHEXIT sollten übrigens die Flags **SELECTABLE** und **EXIT** nicht gesetzt sein.

Im Beispielprogramm wird TOUCHEXIT dazu verwendet, Schieberegler zu realisieren. Diese Schieberegler wurden mit dem RCS konstruiert. Wichtig ist, daß man einen Free Tree benutzt, da man nur dann die Objekte stufenlos vergrößern und verkleinern kann.

Ein Schieberegler besteht im wesentlichen aus drei Teilen: Einer Box, die den Reglerpfad symbolisiert (im Programm FBAR und ABAR), einer Box für den eigentlichen Reglerknopf (FSLIDE und ASLIDE) und einer Bounding Box (FEXT und AEXT), die beide anderen

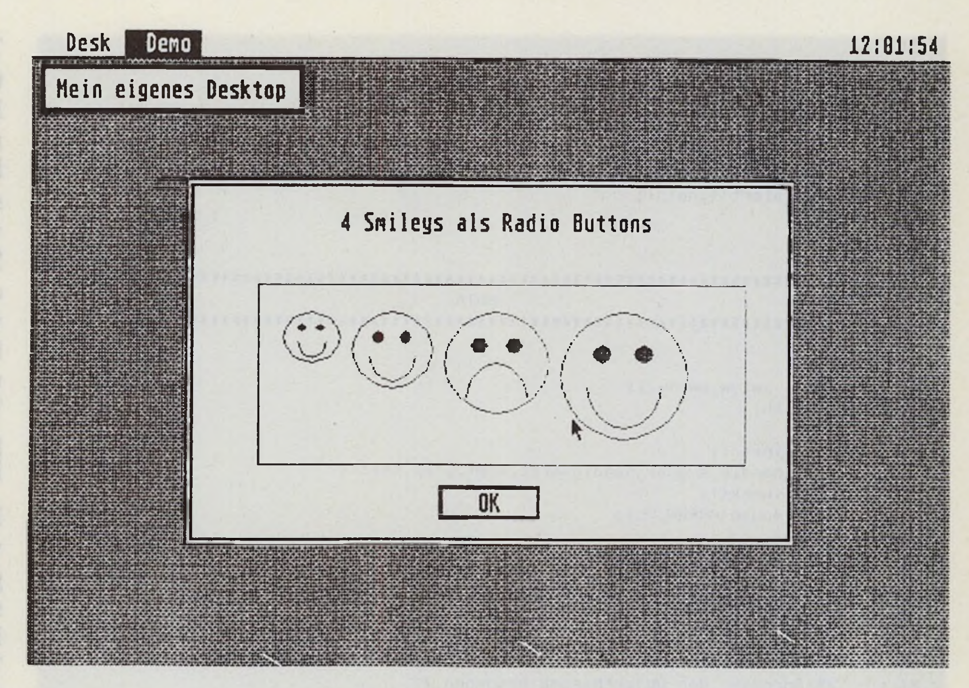


Bild 2

Listing 2

```
 1 : /*
 2 :  * GEMKURS4.C  von Th. Weinstein, Karlsruhe, den 13.1.87
 3 :  *
 4 :  * Demoprogramm fuer erweiterte Objektbehandlung. Anhand von Schiebereglern
 5 :  * zur Frequenzeinstellung des ATARI Soundchip und einem nicht ganz so ernst
 6 :  * gemeinten SMILEY Button werden erweiterte Objektypen wie TOUCHEXIT,
 7 :  * INDIREKT UND USERDEF gezeigt.
 8 :  *
 9 :  */
10 :
11 : /*
12 :  * INCLUDES
13 :  */
14 : #include <define.h>
15 : #include <osbind.h>
16 : #include <gemdefs.h>
17 : #include <obdefs.h>
18 : #include "gemkurs4.h"
19 :
20 : /*
21 :  * TYPEN
22 :  */
23 : typedef unsigned char    BYTE;
24 : typedef struct kanal {
25 :     char *ob_spec;            /* hier wird ob_spec eingetragen */
26 :     int   fpos;               /* Position Frequenz Regler      */
27 :     int   apos;               /* Position Amplituden Regler    */
28 : }  KANALTYP;
29 :
30 : /*
31 :  * EXTERNALS
32 :  */
33 : extern int       phys_handle;
34 : extern int       handle;
35 :
36 : /*
37 :  * GLOBALS
38 :  */
39 : int        ret;
40 : int        quit       = 0;
41 : int        act_kanal  = 0;
42 : KANALTYP   kanal[3];
43 :
44 : OBJECT   *menuaddr;          /* Adresse Menueleistenbaum              */
45 : OBJECT   *infoaddr;          /* Adresse Infodialog                    */
46 : OBJECT   *smileaddr;         /* Adresse Smileydialog                  */
47 : OBJECT   *soundaddr;         /* Adresse Sounddialog                   */
48 : OBJECT   *mydeskaddr;        /* Adresse Objektbaum neues Desktop      */
49 :
50 : APPLBLK  smile_app;          /* Application Block fuer User def. Objekt*/
```

```
 51 :
 52 : message(msg)
 53 : char *msg;
 54 : {
 55 :     char  help[80];
 56 :
 57 :     sprintf(help,"[1][%s][ OK ]",msg);
 58 :     form_alert(1,help);
 59 : }
 60 :
 61 :
 62 : /*************************************************************************/
 63 : /*                                   MAIN                                  */
 64 : /*************************************************************************/
 65 : main()
 66 : {
 67 :     int   xw,yw,ww,hw,i;
 68 :     char *h;
 69 :
 70 :     appl_init();
 71 :     phys_handle = graf_handle(&ret,&ret,&ret,&ret);
 72 :     open_vwork();
 73 :     graf_mouse(ARROW,0L);
 74 :
 75 :     if (!rsrc_load("gemkurs4.rsc"))
 76 :     {
 77 :        message("Wo ist GEMKURS4.RSC");
 78 :        goto ENDE;
 79 :     }
 80 :
 81 :     /* Adressen der Objektbaeume besorgen */
 82 :     rsrc_gaddr(R_TREE,MENU,  &menuaddr);
 83 :     rsrc_gaddr(R_TREE,INFO,  &infoaddr);
 84 :     rsrc_gaddr(R_TREE,SOUND, &soundaddr);
 85 :     rsrc_gaddr(R_TREE,SMILER,&smileaddr);
 86 :     rsrc_gaddr(R_TREE,MYDESK,&mydeskaddr);
 87 :
 88 :     /* Eigenen Desktop Hintergrund installieren */
 89 :     wind_get(0,WF_WORKXYWH,&xw,&yw,&ww,&hw);
 90 :     mydeskaddr->ob_x      =  xw;
 91 :     mydeskaddr->ob_y      =  yw;
 92 :     mydeskaddr->ob_width  =  ww;
 93 :     mydeskaddr->ob_height =  hw;
 94 :     wind_set(0,WF_NEWDESK,mydeskaddr,0,0);
 95 :     form_dial(3,0,0,640,400,0,0,640,400);
 96 :
 97 :     /* User definiertes Objekt einhaengen */
 98 :     fix_smiley(smileaddr,SMILEY1);
 99 :     fix_smiley(smileaddr,SMILEY2);
100 :     fix_smiley(smileaddr,SMILEY3);
101 :     fix_smiley(smileaddr,SMILEY4);
102 :     smileaddr[SMILEY1].ob_state |= SELECTED;   /* Einer muss selektiert sein */
103 :                                                /* wg. Radio Button           */
104 :
105 :     /* Kanalwahl mit INDIRECT Flag initialisieren */
106 :     soundaddr[KANAL].ob_flags |= INDIRECT; /* INDIRECT Flag setzen */
107 :
108 :     for (i=0; i<3; i++)
109 :     {
110 :          kanal[i].ob_spec  = soundaddr[KANAL].ob_spec;
111 :          h = (char *) &kanal[i].ob_spec;
112 :         *h = i + '1'; /* Kanalnummer ins hoechste Byte von ob_spec schreiben */
113 :     }
114 :     soundaddr[KANAL].ob_spec   = (char *) &kanal[0].ob_spec;
115 :
116 :     menu_bar(menuaddr,TRUE);    /* Menuezeile einschalten */
117 :
118 :     multi();
119 :
120 : ENDE:
121 :     close_vwork();
122 : }
123 :
124 :
125 : multi()
126 : {
127 : int   event;
128 : int   msgbuff[8];                     /* Nachrichten Puffer
129 :
130 :
131 :     do {
132 :          event = evnt_multi(MU_MESAG,
133 :                             1,1,0,
134 :                             0,0,0,0,0,
135 :                             0,0,0,0,0,
136 :                             msgbuff,0,0,&ret,&ret,&ret,&ret,&ret,&ret);
137 :      /*  evnt_mesag(msgbuff);  Das reicht hier eigentlich auch !!! */
138 :
```

Teile umschließt. Bei der Konstruktion ist darauf zu achten, daß die BARS und SLIDES direkte Nachkommen der Bounding Boxes im Objektbaum sind. Damit die einmal hergestellte Struktur nicht mehr verändert wird, sollte man das Sortieren bei Schiebereglern vermeiden. Eine einfache Möglichkeit, im RCS die Verwandschaftsverhältnisse festzustellen, ist folgende: Man klickt ein Objekt an und betätigt dabei die CONTROL-Taste. Dadurch wird der direkte Vorgänger des angeklickten Objekts selektiert.

Die Reglerskala ist einfach ein String vom Typ BOXTEXT. Skala und Regler habe ich wiederum in einer umfassenden Box zusammengefaßt, um den Regler als ganzes verschieben (CONTROL Taste) und kopieren (SHIFT-CONTROL Tasten) zu können.

Die Programmierung des Schiebereglers sehen sie in der Funktion **hndl__touchexit()**, Zeile 296-332. Der Regler kann, wie üblich, direkt mit der Maus verstellt werden, indem man entweder auf dem Reglerknopf die Maustaste drückt und dann den Knopf verschiebt, oder (schneller) durch Anklicken des Reglerpfades an der gewünschten Stelle.

Neben ein bißchen Hin- und Herrechnerei mit Objektkoordinaten wird nur noch die Funktion **graf__slidebox()** verwendet. Sie erlaubt es, ein Rechteck in einem anderen zu bewegen, und liefert nach Loslassen der Maustaste einen Wert zwischen 0 und 1000 zurück, der die Stellung der Rechtecke zueinander ausdrückt. Die Rechtecke sind dabei BOX Objekte in einem Objektbaum (im Beispiel bewegt sich z.B ASLIDE innerhalb von AEXT). Leider muß man die Objektkoordinaten selbst verändern (s. **set__slider()**).

Das INDIREKT Flag

Ein weiteres nützliches Flag ist das INDIREKT-Flag. Wenn es gesetzt ist, interpretiert GEM das **ob__spec** Feld einer Objektstruktur nicht als die nähere Spezifikation des Objekts, sondern als Zeiger auf sie. Man kann dies verwenden, um etwa zu einem Objekt eine Struktur mit weiteren Angaben zum Objekt anzulegen. An die erste Stelle dieser Struktur schreibt man den alten **ob__spec** Eintrag, und an Stelle des **ob__spec** schreibt man im Objekt einen Zeiger auf die neue Struktur.


C-soft
GmbH




HANNOVER-MESSE
CeBIT'87

Das Desktop z. B. verwendet diese Methode, um zu jedem Dateiobjekt (Icon oder Text) weitere Einzelheiten zur Datei zu speichern.

In Kombination mit dem TOUCHEXIT-Flag bieten sich weitere interessante Möglichkeiten. Das Beispielprogramm benutzt das INDIREKT-Flag zum Einstellen der Kanalnummer (siehe Bild 1). Bei jedem Anklicken der Kanalnummer wird der **form_do()** Aufruf verlassen (TOUCHEXIT) und der **ob_spec** Eintrag (INDIREKT) auf eine neue Struktur vom Typ KANALTYP gerichtet. In der Struktur „Kanaltyp" ist an erster Stelle der alte **ob_spec** Eintrag gespeichert. Da das Objekt vom Typ BOXCHAR ist, steht im höchsten Byte derjenige Buchstabe, der im Button erscheinen soll. Außerdem stehen in der Struktur die Schiebereglerpositionen für Frequenz und Lautstärke.

Die Initialisierung der Kanalstruktur und des Buttonobjekts finden Sie in Listing 2, Zeile 105-114, ihre Verwendung in der Funktion **handl_touchexit()**, Zeile 288-294.

Ein eigenes Desktop

Eine nette Sache, die nur am Rande mit Objekten zu tun hat, ist das Anlegen eines eigenen Desktophintergrunds. GEM übernimmt ein vom Benutzer definiertes Desktop mit dem Aufruf:

wind_set (0,WF_NEWDESK,tree,0,0);

tree ist dabei die Adresse eines Objektbaums, den man z.B. mit dem RCS konstruieren kann. In diesem Objektbaum ist alles erlaubt, auch benutzerdefinierte Objekte, die ich Ihnen gleich erklären werde. In Listing 2 habe ich als tree einfach eine Dialogbox mit einem dunklen Füllmuster übergeben, die in der linken oberen Ecke einen BOXTEXT beinhaltet. Mit der Größe der Objekte muß man aufpassen, sonst gibt es leicht ausgefranste Ränder. Wie man das Ganze macht, sehen sie in Zeile 88-95 von Listing 2. Das Ergebnis ist auf den Bildern 1 und 2 zu sehen.

Unbedingt notwendig ist der **form_dial(3,...)**-Aufruf, um GEM zu einem kompletten Redraw des Desktops zu veranlassen. Der besondere Gag ist, daß von diesem Moment an GEM das neue Desktop ganz allein verwaltet.

```
139 :         wind_update(TRUE);
140 :
141 :         if ((event & MU_MESAG) && (msgbuff[0] == MN_SELECTED))
142 :         {
143 :           switch (msgbuff[3])  /* Title */
144 :           {
145 :              case MDESK:
146 :                      if (msgbuff[4] == MINFO)
147 :                            hndl_dialog(infoaddr,0);
148 :              break;
149 :
150 :              case MDEMO:
151 :                      switch (msgbuff[4])  /* Item */
152 :                      {
153 :                         case MSOUND:
154 :                              hndl_dialog(soundaddr,0);
155 :                         break;
156 :
157 :                         case MSMILE:
158 :                               hndl_dialog(smileaddr,0);
159 :                         break;
160 :
161 :                         case MQUIT:
162 :                                   quit = TRUE;
163 :                         break;
164 :                      }
165 :              break;
166 :           }
167 :           menu_tnormal(menuaddr,msgbuff[3],1);  /* Menuetitel normalisieren */
168 :         }
169 :
170 :         wind_update(FALSE);
171 :
172 :       } while(!quit);
173 :
174 : } /* multi() */
175 :
176 :
177 : /*
178 :  * Haengt den Smiley an jede gewuenschte Stelle im Baum
179 :  */
180 : fix_smiley(tree,index)
181 : OBJECT *tree;
182 : int      index;
183 : {
184 :       int   draw_smiley();                     /* Funktion bekannt machen     */
185 :
186 :       smile_app.ub_code = draw_smiley;  /* Adresse der Zeichenfunktion */
187 :       smile_app.ub_parm = 0L;                /* wird hier nicht benutzt     */
188 :
189 :       tree[index].ob_flags |= SELECTABLE; /* Sonst tut sich nichts !        */
190 :       tree[index].ob_type   = G_PROGDEF;   /* Typ USER DEFINED               */
191 :       tree[index].ob_spec   = (char *) &smile_app; /* In Objekt einhaengen */
192 :
193 : } /* fix_smiley */
194 :
195 : int draw_smiley(pblk)
196 : PARMBLK *pblk;     /* wird vom Resourcemanager uebergeben */
197 : {
198 :    int        x, y, w, h;         /* Aus reiner Schreibfaulheit */
199 :
200 :    x     = pblk->pb_x;
201 :    y     = pblk->pb_y;
202 :    w     = pblk->pb_w;
203 :    h     = pblk->pb_h;
204 :
205 :    pblk->pb_wc += pblk->pb_xc - 1; /* vs_clip will Eckpunkte des Clip   */
206 :    pblk->pb_hc += pblk->pb_yc - 1; /* rechtecks haben.                  */
207 :    vs_clip(handle,1,(int *) &pblk->pb_xc);
208 :
209 :    vsl_color(handle,1);
210 :    vsf_interior(handle,1);
211 :
212 :    /* Zuerst der Kopf */
213 :    vsf_color(handle,0);
214 :    v_ellipse(handle,x+w/2,y+h/2,w/2,h/2);
215 :    v_ellarc(handle,x+w/2,y+h/2,w/2,h/2,0,3600);
216 :
217 :    /* Jetzt die Augen */
218 :    vsf_color(handle,1);     /* Schwarz wie die Nacht */
219 :    v_ellipse(handle,x+  w/3,y+h/3,w/12,h/12); /* Das Linke */
220 :    v_ellipse(handle,x+2*w/3,y+h/3,w/12,h/12); /* Das Rechte*/
221 :
222 :    /* Zuletzt vom Zustand abhaengig: der Mund */
223 :    if (pblk->pb_currstate & SELECTED)
224 :          v_arc(handle,x+w/2,y+10*h/12,h/3,0,1800);
225 :    else
226 :          v_arc(handle,x+w/2,y+2*h/3-h/12,h/3,1800,0);
227 :
```



Programmieren unter GEM? „Schwierige Einarbeitung und äußerst zeitaufwendig", winkten selbst Insider unlängst ab. Dann kam „Das große GEM-Buch" aus dem Hause DATA BECKER. Ein kompaktes, übersichtlich gegliedertes Buch. Ein Buch voller Lösungen. Es verhalf nicht nur Insidern zum nötigen Durchblick bei der Systemprogrammierung. Und dennoch: Einige wirklich harte Nüsse gab es noch zu knacken. Eine wahre Herausforderung für die Autoren dieses Buches. Sie forschten weiter. Das Ergebnis ihres Ehrgeizes liegt nun vor: ATARI ST GEM. Ein schlichter, präziser Titel für ein Buch, das es in sich hat. Mit einer Fülle an Facts und Informationen zu GEM, wie sie es bisher in ähnlich umfassender Form noch nicht gegeben haben dürfte. Alles, was es zu GEM zu sagen gibt, steht in diesem Buch. Sie erfahren, wie einzelne Funktionen zusammenhängen, nach welchem Konzept GEM aufgebaut ist, was die Systembibliotheken leisten können und vieles mehr. Ist dann genügend Grundwissen vorhanden, legen die Autoren erst richtig los. Systemaufrufe aus GfA-BASIC, C und Assembler, Erstellung eigener GEM-Bindings, Aufbau der Resourcen, Programmierung von Slider-Objekten, Aufbau eines eigenen Desktop und vor allen Dingen: Ein komplett kommentiertes VDI Listing sowie kommentierte Listings ausgewählter AES Funktionen. Alles praxisbezogen und mit vielen Beispielen. Denn ein gut ausgewähltes Listing sagt oft mehr als eine detaillierte Beschreibung. Arbeiten Sie mit diesem Buch, Sie werden vergebens auf die gefürchteten Bömbchen warten. Am Ende liegt Ihnen das gesamte Betriebssystem zu Füßen. Resultat? Nur noch anwenderfreundliche Programme: KLICK.

ATARI ST GEM
Hardcover,
691 Seiten,
DM 69,-

BESTELL-COUPON

Einsenden an: DATA BECKER · Merowingerstr. 30 · 4000 Düsseldorf 1
Bitte senden Sie mir:
☐ ATARI ST GEM DM 69,-
zzgl. DM 5,- Versandkosten
☐ per Nachnahme ☐ Verrechnungsscheck liegt bei

Name
Straße
Ort

DATA BECKER

Merowingerstr. 30 · 4000 Düsseldorf · Tel. (0211) 310010

Also: Was immer Sie übergeben haben – GEM managt alle notwendigen Redraws selbständig. Mit Hilfe von benutzerdefinierten Objekten könnten Sie sich ohne weiteres einen rosa Hintergrund mit blauen Tupfen anlegen, oder ein digitalisiertes Bild Ihrer Freundin oder Ihres Freundes.

Benutzerdefinierte Objekte

Dies ist der letzte große Abschnitt zu den Objektbäumen unter GEM. Wer sich bis hierher vorwagt, kann alles machen: Thermometerskalen, Analoganzeigen, Buttons, die beim Anklicken eine Melodie spielen, Objekte, die ihre Form verändern, aus GEM-DRAW importierte Handzeichnungen usw.

Der Schlüssel zu all diesen Wundern ist der Objekttyp G_PROGDEF. Wenn GEM diesen Typ vorfindet, erwartet es, daß der **ob_spec** Eintrag des Objekts auf eine Struktur vom Typ **APPLBLK** (Application Block) zeigt. Diese Struktur hat folgendes Aussehen:

```
typedef struct applblk {
        int (*ub_code)();
        long ub_parm;
    } APPLBLK;
```

Der erste Eintrag muß ein Zeiger auf eine Funktion sein, die einen **int**-Wert zurückliefert. Der zweite Eintrag ist beliebig und wird beim Aufruf der Zeichenfunktion verfügbar gemacht.

Jedesmal, wenn nun das Objekt gezeichnet werden muß - entweder durch einen Aufruf von **objc_draw()** oder **objc_change()** - wird die angegebene Funktion ausgeführt. Als Parameter erhält sie einen Zeiger auf eine Struktur namens **PARAMBLK**, die so definiert ist:

```
typedef struct parm_blk {
  OBJECT  *pb_tree;
  int      pb_obj;
  int      pb_prevstate, pb_currstate;
  int      pb_x, pb_y, pb_w, pb_h;
  int      pb_xc, pb_yc, pb_wc, pb_hc;
  long     pb_parm;
} PARAMBLK;
```

pb_tree ist der Objektbaum, in dem sich das Objekt befindet, **pb_obj** ist der Index des Objekts. **pb_prevstate == pb_currstate** bedeutet, daß das Objekt gezeichnet werden soll, **!=** heißt, daß der Zustand des Objekts geändert wurde. Die zwei Statevariablen sind so aufgebaut wie das Feld **ob_**

```
228 :     return(0);
229 : }
230 :
231 : /*
232 :  * Managt den gesamten Dialog mit einer Dialogbox, einschliesslich Zeichnen
233 :  * des Dialogs und Wiederherstellen des Bildschirms am Schluss.
234 :  * Erweitert um die Behandlung von TOUCHEXIT Objekten
235 :  *
236 :  * EINGABE:
237 :  * addr ist die Adresse eines Dialogbaums
238 :  * edit_item ist der Index des ersten editierbaren Textfeldes im Baum und
239 :  * 0 falls keines vorhanden.
240 :  *
241 :  * AUSGABE:
242 :  * exit_cond des Dialogs, d.h Index des Exit Buttons
243 :  */
244 : int   hndl_dialog(addr,edit_item)
245 : OBJECT   *addr;
246 : int      edit_item;
247 : {
248 :     int   ex_cond;
249 :     int   x,y,w,h;
250 :
251 :     form_center(addr,&x,&y,&w,&h);
252 :
253 :     form_dial(FMD_START,318,198,4,4,x,y,w,h);
254 :     form_dial(FMD_GROW,318,198,4,4,x,y,w,h);
255 :
256 :        objc_draw(addr,0,MAX_DEPTH,x,y,w,h);
257 :     do
258 :     {
259 :         ex_cond = form_do(addr,edit_item) & 0x7fff; /* High Bit maskieren */
260 :
261 :         addr[ex_cond].ob_state &= ~SELECTED;
262 :
263 :         /* Spezialbehandlung fuer Touchexitobjekte */
264 :         if (addr[ex_cond].ob_flags & TOUCHEXIT)
265 :            hndl_touchexit(addr,ex_cond,x,y,w,h);
266 :
267 :     } while (addr[ex_cond].ob_flags & TOUCHEXIT);
268 :
269 :     form_dial(FMD_SHRINK,318,198,4,4,x,y,w,h);
270 :     form_dial(FMD_FINISH,318,198,4,4,x,y,w,h);
271 :
272 :     return(ex_cond);
273 :
274 : } /* hndl_dialog */
275 :
276 : hndl_touchexit(addr,item,x,y,w,h)
277 : OBJECT *addr;
278 : int    item;
279 : int    x,y,w,h;
280 : {
281 :     int xoff, mx;
282 :     int slidepos;
283 :     int we, ws;
284 :
285 :     if (addr == soundaddr)
286 :        switch (item)
287 :        {
288 :           case KANAL:
289 :                 act_kanal = (act_kanal + 1) % 3;
290 :                 soundaddr[KANAL].ob_spec = (char *) &kanal[act_kanal].ob_spec;
291 :                 set_slider(soundaddr,FEXT,FSLIDE,kanal[act_kanal].fpos);
292 :                 set_slider(soundaddr,AEXT,ASLIDE,kanal[act_kanal].apos);
293 :                 objc_draw(addr,0,MAX_DEPTH,x,y,w,h);
294 :           break;
295 :
296 :           case FBAR:
297 :                 objc_offset(soundaddr,FEXT,&xoff,&ret);
298 :                 graf_mkstate(&mx,&ret,&ret,&ret);
299 :                 mx -= xoff;
300 :                 we = soundaddr[FEXT].ob_width;
301 :                 ws = soundaddr[FSLIDE].ob_width;
302 :                 if (mx > (we-ws)) mx = we-ws;
303 :                 kanal[act_kanal].fpos = (1000L * (long) mx) / (long)(we-ws);
304 :                 soundaddr[FSLIDE].ob_x = mx;
305 :                 objc_draw(addr,FEXT,MAX_DEPTH,x,y,w,h);
306 :           break;
307 :
308 :           case ABAR:
309 :                 objc_offset(soundaddr,AEXT,&xoff,&ret);
310 :                 graf_mkstate(&mx,&ret,&ret,&ret);
311 :                 mx -= xoff;
312 :                 we = soundaddr[AEXT].ob_width;
313 :                 ws = soundaddr[ASLIDE].ob_width;
314 :                 if (mx > (we-ws)) mx = we-ws;
315 :                 kanal[act_kanal].apos = (1000L * (long) mx) / (long)(we-ws);
316 :                 soundaddr[ASLIDE].ob_x = mx;
```

```
317 :                 objc_draw(addr,AEXT,MAX_DEPTH,x,y,w,h);
318 :         break;
319 :
320 :         case FSLIDE:
321 :                 slidepos = graf_slidebox(soundaddr,FEXT,FSLIDE,0);
322 :                 kanal[act_kanal].fpos = slidepos;
323 :                 set_slider(soundaddr,FEXT,FSLIDE,slidepos);
324 :                 objc_draw(addr,FEXT,MAX_DEPTH,x,y,w,h);
325 :         break;
326 :
327 :         case ASLIDE:
328 :                 slidepos = graf_slidebox(soundaddr,AEXT,ASLIDE,0);
329 :                 kanal[act_kanal].apos = slidepos;
330 :                 set_slider(soundaddr,AEXT,ASLIDE,slidepos);
331 :                 objc_draw(addr,AEXT,MAX_DEPTH,x,y,w,h);
332 :         break;
333 :     }
334 :
335 : } /* hndl_touchexit() */
336 :
337 : set_slider(addr,ext,slide,pos)
338 : OBJECT *addr;
339 : int     ext,slide,pos;
340 : {
341 :     int we, ws;
342 :
343 :     we                 = addr[ext].ob_width;
344 :     ws                 = addr[slide].ob_width;
345 :     addr[slide].ob_x = ((long)(we - ws) * (long) pos) / 1000L;
346 : }
```

state in der Objektstruktur. **pb__x** .. **pb__h** sind die Abmessungen der Bounding Box des Objekts und **pb__xc** .. **pb__hc** ist das Cliprechteck, in dem gezeichnet werden soll. **pb__parm** ist identisch mit **ub__parm aus dem APPLBLK.** Mit diesen Informationen kann die Zeichenfunktion ihren Job erledigen.

Die Struktur **APPLBLK** können Sie übrigens beliebig erweitern, da der benötigte Speicherplatz von Ihrem Programm reserviert wird. Die **PARAMBLK** Struktur darf nicht erweitert werden, da sie im Datenbereich des AES angelegt wird.

Innerhalb der Zeichenroutine gilt eine strikte Regel: Es dürfen keine AES-Aufrufe gemacht werden! VDI-Aufrufe sind dagegen erlaubt. Das kommt daher, daß Ihre Zeichenroutine von AES ausgeführt wird und AES selbst nicht reentrant (wiedereintrittsfähig) ist. Sie sollten auch darauf achten, Parameterlisten nicht zu lang zu machen oder viele verschachtelte Funktionsaufrufe zu vermeiden, da die Zeichenroutine den (sehr kleinen !?) Stackbereich des AES benutzt.

Als Beispiel habe ich Ihnen einen Smiley Button programmiert. Wie er aussieht, können Sie auf Bild 2 bewundern; wie er programmiert ist, in der Funktion **draw__smiley()** von Listing 2. Nicht selektiert zeigt er ein fröhliches Gesicht, selektiert ein böses.

Dabei können Sie gleich ein Problem von runden Objekten feststellen: Es gibt Stellen außerhalb des Objekts, die selektierend wirken. Es sind genau die Bereiche zwischen dem Kreis und seiner Bounding Box.

Und nun: Viel Spaß beim Entwerfen von eigenen Objekten, Desktops und allen möglichen Kombinationen!

Th. Weinstein

MPROLOG

jetzt auch für den ATARI ST

Was ist MProlog?

Der Name MProlog steht für Modular Prolog. MProlog ist das erste kommerzielle Prolog-Entwicklungssystem, das für den ATARI ST erhältlich ist. Für Insider und KI-Spezialisten ist MProlog sicher ein Begriff. Für alle anderen ein kurzer Überblick:

MProlog, eine ursprünglich ungarische Entwicklung, wurde in der leider nicht sehr verbreiteten Compilersprache CDL (Compiler Design Language) geschrieben. CDL-Compiler erzeugen einen hoch-optimierten Code und geben dem Entwickler alle Möglichkeiten, das Letzte aus einem Rechner herauszuholen. In CDL wurden zum Beispiel die Programme der bekannten „Mephisto"-Schachcomputer geschrieben.

MProlog wurde bereits an viele Rechner angepaßt: Von kleinen Geräten, wie IBM PC, Apple Macintosh und jetzt Atari, über Sun, Apollo, VAX und Nixdorf bis hin zu Großrechnern der Marken IBM und Siemens. MProlog kann weltweit mehr als 1000 Installationen in Industrie und Wissenschaft aufweisen.

Die Programme sind bis auf wenige rechnerspezifische Details zwischen den einzelnen Implementierungen auf Quellcodeebene vollständig portabel. Programme, die auf einem günstigen System wie dem Atari ST entwickelt wurden, laufen in ihrer Produktionsversion ohne weiteres auf Rechnern wie VAX oder IBM ab.

Der Interpreter

Kernstück des Systems ist, wie bei allen Prolog Implementierungen, der Interpreter. Er arbeitet die Programme ab und spuckt die erwarteten Ergebnisse aus. Der MProlog Interpreter erwartet aus Effizienzgründen eine vorverarbeitete Eingabe. Das macht sich dann auch in der Geschwindigkeit bemerkbar: Mit einem mitgelieferten Testprogramm erhält man Werte zwischen 600 und 800 Lips (s. Kasten).

Was ist ein Lips?

Lips steht für Logical Inferences per Second, das ist: die Anzahl von logischen Schlüssen pro Sekunde. Für die Messung gibt es anscheinend keine bindende Vereinbarung. Meist wird das folgende **naive_reverse** genannte Programm zur Messung benutzt. Es kehrt eine gegebene Liste um. Die Anzahl der Aufrufe von **naive_reverse**, die der Interpreter in der Sekunde leistet, ist dann genau die gesuchte Lipszahl.

```
naive_reverse([X|L0],L) :-
                naive_reverse(L0,L1),
                append(L1,[X],L).
naive_reverse([],[]).
```

MProlog hält sich rein syntaktisch an den DEC-10-Standard, weicht jedoch bei der Namensgebung der Prädikate vollkommen davon ab. Das mag den, der bereits Prologerfahrung hat, etwas stören. Nach einer raschen Gewöhnung möchte man die gegenüber dem DEC-10-Standard bessere Gliederung und Vereinheitlichung nicht mehr missen. Für den eingefleischten DEC-10-Fan steht sogar eigens ein Satz an Prädikaten zur Verfügung, die eine Umsetzung von DEC-10 nach MProlog ermöglichen.

Von vornherein enthält MProlog einen gewaltigen Satz an eingebauten Prädikaten aus allen Sparten (ca. 150 Standardprädikate). Tabelle 1 zeigt einen willkürlichen, alphabetischen Ausschnitt.

Die eingebauten Prädikate ermöglichen dem Benutzer z. B. die vollständige Kontrolle über die dynamische Datenbasis. Er kann global oder ganz selektiv einfügen, löschen und zugreifen. Die meisten Prädikate, die die Datenbasis beeinflussen, sind wahlweise auch rücksetzbar: Bei einem Rücksetzen im Suchbaum macht der Interpreter die ausgelösten Effekte wieder rückgängig. Welche Möglichkeiten diese Option bietet, muß ich wohl nicht erklären.

Sehr selten bei Prolog ist der hier vollständig vorhandene Satz an Prädikaten zur Stringbehandlung. Strings können verkettet werden; es ist möglich, Substrings zu suchen und Strings zu zerteilen. Es gibt ein Prädikat, das die Länge eines Strings liefert und vieles mehr. Außerdem ist es möglich, einen String in eine Liste zu verwandeln; damit stehen alle listenverarbeitenden Prädikate zur Verfügung.

Wichtig für eine Programmiersprache ist das Ein-/Ausgabehandling. Auch hier hat MProlog einiges zu bieten. Die Ein-/Ausgabe ist ähnlich wie in C Stream-orientiert. Das heißt, man ordnet einem Ein- oder Ausgabestrom eine Datei oder ein physikalisches Gerät zu und kann dann einzelne Zeichen, Token oder ganze Statements lesen und schreiben. Wie bei der Datenbasis sind die meisten Prädikate rücksetzbar. Beim Backtracking z. B. werden dadurch bereits gelesene Zeichen in die Eingabe zurückgestellt. Außerdem gibt es spezielle Prädikate zur Abfrage der Tastatur und zur Ausgabe auf den

Tabelle 1: Standardprädikate (Ausschnitt)

!	/	+	-
->	=/=	/=	<
<	accept_char	accept_key	add_op
add_statement	ancestor	ancestor_list	and
bag_of	channel_state	close_input	compare
concatenate	convert	create_*	cursor
cut_input	del_all_statements	del_op	fail
is_a	key_sort	list_length	modify
output_columns_left	phrase	protected	read
read_token	remove_spaces	set_cursor	set_of
signal	sort	string_index	system
unread_token	value	var_list	write
usw.			

Bildschirm.

Schließlich sollen die Möglichkeiten zur Fehlerbehandlung erwähnt werden. Jedes Prädikat, z. B. auch das ganze Programm, kann in einem protected- (geschützten) Modus aufgerufen werden. Tritt dann ein Fehler auf, so wird ein vom Benutzer definiertes Prädikat aufgerufen, das die Entscheidung über den weiteren Verlauf treffen kann. Diese sogenannten Error-Handler werden auf einem Stack abgelegt, so daß ein nicht abgefangener Fehler an den nächsten Handler weitergereicht wird. Dadurch ist es möglich, eine ausgeklügelte Hierarchie von Fehlerlerbehandlungsprädikaten anzulegen und so eine vollständige Kontrolle des Programms im Fehlerfall zu erhalten.

Der Interpreter bietet darüber hinaus noch Möglichkeiten, bei der Definition eines Prädikats die Reihenfolge festzulegen, in der bei einem späteren Aufruf die Argumente gematcht werden. Weitere Effizienzsteigerungen sind durch die Kennzeichnung von immer gebundenen Variablen möglich. Das weitere Verhalten des Interpreters ist durch eine Menge von Optionen steuerbar.

Für den Insider sei angemerkt, daß MProlog auch die üblichen Prädikate zur Verarbeitung von Definite Clause Grammars beinhaltet und damit die Programmierung von natürlichsprachlichen Interfaces stark erleichtert.

Die Graphik

Das MProlog Paket für den Atari schließt eine Grafik ein. Ein Atari-Freak, der erwartet, so einen Durchgriff auf VDI und AES zu haben, muß allerdings enttäuscht werden: Die Grafik selbst benutzt zwar VDI-Aufrufe und erreicht damit die vom Atari gewohnte Geschwindigkeit, der Benutzer hat aber keinen direkten Zugriff. Das hat natürlich seinen Grund: Oberstes Gebot bei MProlog ist, wie schon erwähnt, die Portabilität von Programmen. Durch das Verbot eines direkten Durchgriffs auf Systemebene wird diese Portabilität momentan zwischen den Rechnern IBM PC, Macintosh und Atari erreicht.

Die Grafik selbst beinhaltet dennoch eine interessante Philosophie: Es ist eine von Logo her bekannte Turtle-Grafik – aber in 3D. Statt der Logo-Schildkröte kann man sich einen Adler vorstellen, der durch den Raum fliegt und dabei eine Farbspur zurückläßt. Man kann den Adler vorwärts und rückwärts, nach rechts und nach links fliegen lassen. Man kann ihn aber auch zu einem Sturzflug veranlassen, oder ihn auf die Seite legen. Die dadurch erzeugte dreidimensionale Flugbahn wird dann vom Sytem wahlweise parallel oder perspektivisch auf den Bildschrim projeziert. Darüber hinaus ist es möglich, Gebiete zu füllen und Text in die Grafik zu schreiben. Es ist sogar ein rudimetärer Verdeckungsalgorithmus vorhanden, der allerdings nur für einen konvexen Körper funktioniert. Zahlreiche Demos zur Grafik werden mitgeliefert. So kann man sich die bekannten Türme von Hanoi als immer kleiner werdende Quader auf dem Bildschirm umschichten lassen.

Modularität

Vielleicht haben Sie sich schon gefragt, was das Wort modular in MProlog bedeutet. Die Erklärung ist einfach: MProlog bietet die Möglichkeit, Programme in einzelne Module aufzuteilen, wie es etwa in Modula 2 gemacht wird. Ein Modul hat ein Import bzw. Export-Interface, in dem angegeben wird, was außerhalb des Moduls sichtbar ist. Dadurch ist es möglich, die eigentliche Implementierung eines Prädikats nach außen hin zu verbergen und nur eine Schnittstelle für die Benutzung zu bieten.

Es ist bekannt, daß die Aufteilung in Module einen großen Einfluß auf die Wartbarkeit und Änderungsfreundlichkeit von großen Programmsystemen hat. Ein Beispiel für das äußere Erscheinungsbild von MProlog-Programmen sehen sie in Listing 1. Es ist ein Programm, das Strings, die auch die Wildcards '*' und '?' enthalten dürfen, gegeneinander matcht.

Was ist Logic Lab?

Logic Lab heißt Zusammenfassung und Zusammenspiel der Programme im MProlog-Paket. Schauen Sie sich dazu bitte Bild 1 an.

Ganz links oben sehen sie das PDSS (Program Development Support System). Es stellt die Verbindung zwischen Anwender und Interpreter während der Entwicklungsphase eines Programms her. PDSS legt sich wie ei-

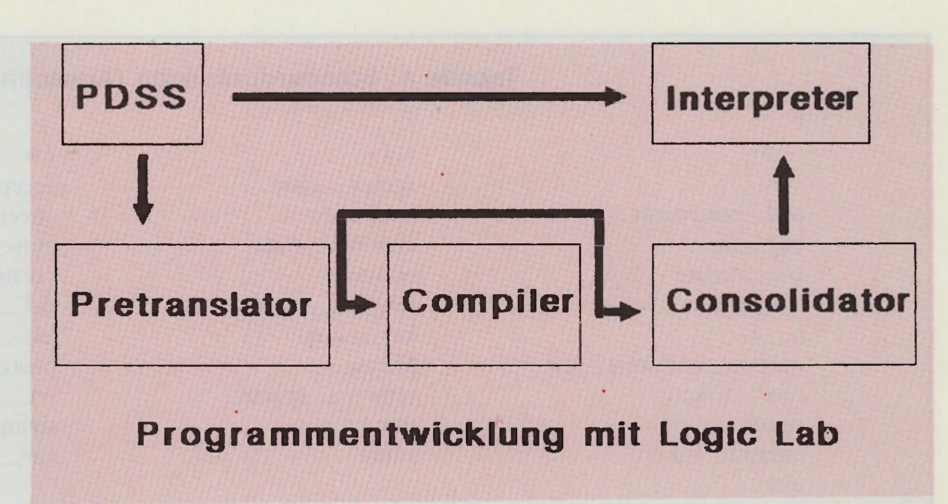
ne Schale um den Interpreter und übersetzt die Benutzeranfragen an den Interpreter. Integriert ist ein Full Screen Editor, der bei Programmierfehlern automatisch aufgerufen wird und den Cursor genau an die Fehlerstelle positioniert. Dadurch sind syntaktische Fehler sehr schnell zu beheben. PDSS stellt eine Reihe mächtiger Kommandos zum Verwalten und Testen eines Programmsystems zur Verfügung. Es ist möglich, die Benutzersicht auf Modulebene und sogar Definitionsebene zu „fokussieren" und damit gezielt Definitionen innerhalb oder zwischen Modulen zu verschieben und zu verändern. An allen vernünftigen Stellen ist '★' als Wildcard möglich:

focus Module ★ Definition test Statement 3 oder kürzer f M ★ D test S 3

durchsucht alle Module nach der Definition test und richtet den Fokus auf das dritte Statement innerhalb der Definition.

Weiterhin steht ein sehr mächtiger interaktiver Trace zur Verfügung, dessen Benutzung jedoch gerade am Anfang wegen der vielen Möglichkeiten etwas schwierig ist. Es ist möglich, Breakpoints zu setzen oder ein Programm im Einzelschritt zu durchlaufen. Man kann den Interpreter zwingen, ein Prädikat mit succeed oder fail zu verlassen und hat so den Programmfluß voll in der Hand. Es ist möglich, aus dem Trace heraus beliebige PDSS-Kommandos zu geben und sogar einen neuen interaktiven PDSS-Level aufzumachen, um sich nach der Beendigung mit 'bye' wieder an der alten Stelle im Programm zu finden. Die Menge der angezeigten Information läßt sich genau angeben, so daß man an weniger interessanten Stellen nicht von einem „Wust" von Details überschüttet wird. Der Trace-Mechanismus wird außerdem bei jedem Programmfehler automatisch aktiviert, dabei wird die Fehlerstelle angezeigt.

Hat man sein Programm intensiv ausgetestet, kann eine Standalone-Version erzeugt werden. Dazu übersetzt man die einzelnen Module mit einem **Pretranslator** (Vorübersetzer) genannten Programm und linkt sie dann mit dem **Consolidator** zusammen. Nach dem



Programmentwicklung mit Logic Lab

```
/*************************************************************************/
/* Realisiert einen endlichen Automaten fuer regulaere Ausdruecke mit    */
/* den Metazeichen * und ?. * steht dabei fuer eine beliebige Anzahl von */
/* Zeichen und ? fuer genau ein Zeichen                                  */
/*                                                                       */
/* erstellt von Th. Weinstein                                            */
/* System: MProlog Logic Lab                                             */
/*************************************************************************/

export(pattern_matching/2).

.body.

pattern_matching(PATT,STR) :-
     PATT=(*), + ;                 % Sonderfall 1: * allein matcht alles
     STR is_a var, - ;             % Sonderfall 2: Wenn Str eine ungebundene
                                   %               Variable ist schlaegt das
                                   %               Praedikat fehl.
     PATT=STR, + ;                 % Sonderfall 3: Wenn Pattern = String ist
                                   %               alles klar: Erfolg melden.

     % Umwandlung in Zeichenlisten zum Vergleich, bei Misserfolg hierher
     % nicht zuruecksetzen.

     make_char_list(PATT,PLIST), make_char_list(STR,SLIST), /,
     match(PLIST,SLIST) .          % Let's match !!!

% Zwei leere Listen matchen immer

match([],[]) :- + .

% Wenn die beiden ersten Zeichen gleich sind hat die Anfrage Erfolg genau
% dann, wenn sich die Restlisten matchen lassen.

match([EL|PTAIL],[EL|STAIL]) :-  match(PTAIL,STAIL) .

match([*],SLIST) :- + .  % * matcht alles was da noch sein mag

% Steht ein * am Anfang des Musters, muß das nachfolgende Zeichen im
% String gesucht werden. Danach muessen der Rest des Musters und der String
% matchen. Falls sich kein erfolg zeigt hat man u.U noch nicht das laengste
% Muster gefunden -> nach Ruecksetzen weitersuchen.

match([*,NEXT|PTAIL],SLIST) :-
     search(NEXT,SLIST,REST),
       (match(PTAIL,REST)
        ;
        match([*,NEXT|PTAIL],REST)
       ).

% Ein ? matcht jedes Zeichen, aber nur genau eines.

match(["?"|PTAIL],[_|STAIL]) :-
     match(PTAIL,STAIL) .

% Sucht das erste Vorkommen von N in einer Liste und liefert den Rest
% der Liste (alles nach N) zurueck.

search(N,[N|R],R) :-      / .
search(N,[_|T],R) :-      search(N,T,R) .
search(_,[],[])   :-      /, fail .
```

Pretranslator kann man optional den **Compiler** starten, der eine wesentliche Erhöhung der Ausführungsgeschwindigkeit bewirkt. Das durch den Consolidator erzeugte Programm kann dann direkt dem Interpreter zur Ausführung übergeben werden.

Auch in der Standalone-Version braucht man nicht auf die Testmöglichkeiten zu verzichten: Es ist möglich, PDSS zum Programm dazuzulinken.

Atari-Spezifika

Die Atari-Version von MProlog wird unterstützt von den Möglichkeiten, die GEM bietet. Interpreter, Pretranslator und Consolidator sind in GEM eingebunden und werden wie gewohnt über die Menüleiste bedient. Es werden drei Fenster geöffnet: Ein Ein-/Ausgabefenster, ein Editorfenster und ein Grafikfenster.

Weiterhin gibt es einige spezielle Prädikate (nicht portabel):

Der Aufruf

answer ("Titelzeile",X).

öffnet eine Dialogbox mit einer Textzeile. Der vom Benutzer eingegebene Text wird an die Variable X gebunden.

Mit

select__menu__item(X,"Titel", ["Eintrag1","Eintrag2"]).

wird ein Eintrag in der Menüleiste erzeugt und das selektierte Item an X gebunden.

An allen Stellen (auch in einem Programm), an denen Dateinamen verwendet werden, wird – falls der Name Wildcards enthält – automatisch die bekannte Atari-File-Selektorbox geöffnet.

Nach soviel Erfreulichem sollen auch einige Nachteile zur Sprache kommen. Der größte dürfte der bekannte Speicherhunger von Prolog sein. MProlog macht da keine Ausnahme. Die Herstellerfirma gibt an, daß dem Interpreter in der Zusammenstellung Control-accessory und Festplatte noch 606 KB Speicher zur Verfügung stehen. Dabei darf man jedoch nicht vergessen, daß während der Entwicklungsphase davon noch der Anteil von PDSS abgezogen wird. Für kleinere und mittlere Anwendungen mag der Platz noch ausreichend sein, wer aber ein Expertensystem schreiben will, sollte seinen Rechner doch möglichst auf 2 MByte aufstocken. Zu bedenken ist auch, daß Prolog als Interpretersprache mit dynamischen Datenstrukturen von Zeit zu Zeit eine Garbage-Collection durchführen muß. Je weniger Speicher vorhanden ist, um so öfter wird dieser Vorgang erforderlich.

```
endmod /* pattern_matching */  .

/****************************************************************************/

module main.

import(pattern_matching/2).

body.

test :-
   repeat,
     write("Bitte geben Sie ein Muster ein (e = Ende): "), nl,
     read_record(Pattern),
     (Pattern = "e", +
     ;
        write("Bitte geben Sie den String der gmatcht werden soll ein: "), nl,
        read_record(String),
        (pattern_matching(Pattern,String),
              write("Match war erfolgreich !"), nl
        ;
              write("Keine $153bereinstimmung"), nl
        ),
        /, fail
     ).

:- test. /* Von PDSS aus kann das Programm mit RUN gestartet werden  */
         /* Wenn es übersetzt wird, startet es automatisch an dieser */
         /* Stelle.                                                  */

endmod /* main */
```

Ein weiterer kleiner Minuspunkt ist die Dokumentation. Trotz des Umfangs von fast 700 Seiten ist sie an manchen Stellen etwas knapp ausgefallen und für Anfänger auf jeden Fall völlig ungeeignet. Zudem muß man sich seine Information teilweise recht mühsam zusammensuchen. Bisweilen mangelt es an Beispielen. Die Dokumentation wird dadurch eher zu einem Nachschlagewerk für den erfahrenen Benutzer. Die Firma epsilon liefert allerdings auf Wunsch ein „Primer" genanntes Paket, daß sehr ausführlich und auch für den Anfänger verständlich in Prolog einführt. Dieser Primer ist allerdings im Standardlieferumfang nicht enthalten.

Zusammenfassung

MProlog ist ein durch und durch professionelles Programmpaket und zielt als solches auch auf den professionellen Markt. Allein der Preis verbietet eigentlich Hobbyanwendungen. MProlog enthält alles, was zu einer Entwicklung von großen Programmsystemen im Bereich der KI notwendig ist. epsilon stellt dem Anwender außerdem Hilfen in Form von Wartungsverträgen und vierteljährlichen Seminaren in Berlin und Karlsruhe zur Verfügung.

MProlog auf Atari, IBM oder Macintosh stellt im Moment, trotz des Preises, die weitaus günstigste Möglichkeit zu professionellen, portablen Entwicklungen im Bereich der künstlichen Intelligenz dar. Allerdings sollte man seinem Rechner soviel Speicher wie möglich zur Verfügung stellen.

Programm:
MPROLOG – modulares Prolog + Entwicklungsumgebung.

Version:
Release 2.1 ab ca.. Juni/Juli 87, dann 2.2.

Lieferumfang:
2 Disketten 720 KB (wahlweise 3 a 360

KB) mit Interpreter, Pretranslator, Consolidator, PDSS Entwicklungssystem und Beispielprogrammen. Ordner mit ca. 700 Seiten Dokumentation (englisch).

Anforderungen:
Atari ST mit 1 MByte RAM, ROS im ROM, 1 einseitiges Laufwerk. Empfohlen: Speicher >1MByte und Festplatte.

Datentypen:
Strukturen, Strings, Listen.
Integer: $-2^{23} - 1$ bis 2^{23}
Real: (64 Bit IEEE)
$-1.8 \star 10^{308} - 1.8 \star 10^{308}$

Geschwindigkeit:
Interpreter: 800 Lips
(Logical Inferences per s)
Compiler: 15 Klips werden erwartet.

Preis:
998,– Release 2.1 ohne Compiler
1.498,– Release 2.2 mit Compiler (ab Juni/Juli 87)

Bezug:
epsilon
Gesellschaft für Softwaretechnik und Systementwicklung, Kurfürstendamm 188/189, D-1000 Berlin 15,
Tel. 0 30 / 882 69 91

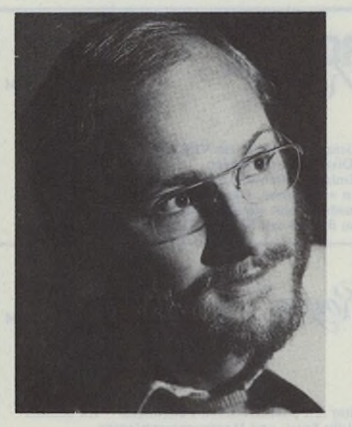
Th. Weinstein



Eine Einführung in Programmiermethoden und Sprachen der KI

1. Teil: Atome und Listen

Der Autor



Dr. Karl Sarnow

wird Sie mit dieser Serie in die Welt der künstlichen Intelligenz einführen.

Er studierte Physik an der TU Hannover und unterrichtet zur Zeit an einem Gymnasium die Fächer Mathematik, Physik und Informatik.

Anfang 1986 stieg er auf den ST um und kommuniziert mit ihm in allen nur erdenklichen Sprachen. Seine bevorzugten Programmiersprachen sind C, LISP und Prolog.

Zum Geleit

Dieser Artikel ist der erste in einer Reihe von Aufsätzen, in denen ich Ihnen Probleme und Verfahrensweisen, die im Bereich der künstlichen Intelligenz vorliegen, nahebringen möchte. Dabei werde ich mich bemühen, die beiden aktuellen Sprachen der KI – LISP und PROLOG – in gleicher Weise zu berücksichtigen. Da beide Sprachen im Public-Domain-Service der ST-Computer-Redaktion vorliegen, sollte es dem Leser nicht schwerfallen, diesem Kurs zu folgen. Übrigens hat jeder ST-Besitzer ja schon von Haus aus die ausgezeichnete Version einer KI-Sprache mitbekommen: Die (viel zu wenig beachtete Logo-Version von Digital Research, die ich bei dieser Gelegenheit gerne aufwerten möchte. LOGO ist nicht **nur** die Turtle-Sprache für Kinder. Als LISP-Derivat ist LOGO ohne Schwierigkeiten in der Lage, das zu vollbringen, was LISP kann. Deshalb werde ich die entsprechende LOGO-Variante ebenfalls erläutern. Jeder Kurs ist einem Schwerpunkt gewidmet und behandelt die für das entsprechende Thema geeigneten Funktionen (LISP, LOGO) bzw. Prädikate (PROLOG). Für eine Übersicht über LISP und PROLOG und die allgemeinen Anwendungsgebiete der KI sei auf meine bisher in dieser Zeitschrift erschienenen Artikel verwiesen (Heft Nr. 1/86 und 3/86). Natürlich wird es sich nicht vermeiden lassen, daß das eine oder andere Thema vorrangig in einer der drei Sprachen behandelt wird und die anderen Sprachen etwas zurücktreten. Ich möchte jedoch den Überblick über die Methoden der KI von der Einengung der jeweils verwendeten Sprache befreien und andererseits die spezifischen Eigenschaften der jeweiligen Sprache erhalten.

Atome und Listen als Datenelemente

Bei Problemen der künstlichen Intelligenz tritt die Rechenfertigkeit hinter der Fähigkeit der Symbolmanipulation zurück. Logischerweise bedient man sich hierzu anderer Datenstrukturen als für die Bearbeitung arithmetischer Probleme. Die beiden Datentypen, mit denen wir uns hauptsächlich beschäftigen, sind **Atome** und **Listen** (einzige Ausnahme: Objekte in XLISP, siehe Literaturliste 2. Atome (gr.: atomos, unteilbar) sind Datenele-

mente, die nicht weiter strukturiert sind und folglich auch nicht in kleinere Bestandteile zerlegt werden können. Dies können symbolische Namen sein (hallo, auto, champagner, hexamethylentetramin, ...) oder Zahlen. Im letzten Fall spricht man von numerischen Atomen.

Listen sind im einfachsten Fall nicht anders als miteinander verbundene Atome. In LISP werden die einzelnen Elemente einer Liste von runden Klammern eingeschlossen, in PROLOG und LOGO von eckigen. Also ist

(hallo auto 3.141)
bzw.
[hallo, auto, 3.141]

eine Liste in LISP, LOGO bzw. PROLOG, welche die Atome ***hallo, auto*** und ***3.141*** enthält. Allerdings können Listen wiederum Listen als Elemente enthalten [(Rekursion, ick hör dir trapsen!)]. Ein etwas sinnvolleres Beispiel wäre (LISP-Notation):

(rufname (vorname1 vorname2 vorname3) name)

Auf die natürliche Intelligenz des Lesers vertrauend, verzichte ich auf die PROLOG- und LOGO-Schreibweise.

In dieser Liste werden die Patennamen als Unterliste geführt. Die Liste selbst besteht also aus drei Elementen, den Atomen *rufname* und *name* sowie der Liste mit den drei Patennamen. LISP schießt in Sachen Listenbehandlung den Vogel ab, weil der Interpreter jeden Befehl in Listenform erwartet.

Die einfachsten Beispiele für Listenverarbeitung stammen deshalb in LISP aus dem Bereich der Arithmetik – ein Indiz für den Wahrheitsgehalt der Vermutung, daß reine Rechenfähigkeit und Intelligenz wenig korreliert sind.

Gibt man in LISP die folgende Liste ein, interpretiert sie der Interpreter als einen auszuführenden Befehl und gibt prompt das Ergebnis der Ausführung bekannt (fett: Benutzereingabe, kursiv: Antwort des Interpreters).

(+ 1 2)
3

Da XLISP ein eval_LISP ist (siehe Literaturliste 2), erkennt das System beim Versuch, die Liste zu evaluieren, daß das erste Element der Liste ein Atom ist, das die Bedeutung einer im System eingebauten Funktion hat, und führt die Funktionsberechnung durch. Solche im System eingebauten Funktionen tragen die Bezeichnung Primitiv. Am Beispiel der Addition in LISP kann man auch einen Vorteil der Liste gegenüber anderen Datenstrukturen erkennen. So erlaubt LISP mehr als 2 Argumente bei allen arithmetischen Operationen:

(+ 1 2 3 4 5)
15

(* 2 4 3)
24

Listen in den KI-Sprachen haben zudem den Vorteil, daß ihre Elemente typenlos sind (im Gegensatz zu PASCAL oder C). Beispielsweise werden die folgenden beiden Listen gleichwertig als Einträge einer Gehaltsliste akzeptiert:

(schmidt emil 1200.50)
(meier (hans georg gustav) 3179.20)

In allen drei Sprachen werden übrigens die Rechenoperationen in Präfix-Notation angegeben. Nur LOGO als „Anfängersprache" hat hier dem traditionellen Infix-Denken Tribut gezollt und erlaubt zusätzlich noch die Infix-Notation für arithmetische Operationen. Die Präfix-Notation hat den Vorteil, daß der Operator immer an der gleichen Stelle steht, unabhängig von der Anzahl der Argumente der Operation. Das Umdenken ist zwar lästig, doch wenn die natürliche Intelligenz beim Umgang mit der künstlichen Intelligenz etwas abbekommt, so ist das ein nicht nur erwünschter, sondern beabsichtigter Nebeneffekt.

Zugriff auf Listenelemente

Der Zugriff auf die Listenelemente ist natürlich das A und O der Programmierung in unseren drei Lieblingssprachen. Tab. 1 faßt die elementaren Zugriffsmöglichkeiten in diesen Sprachen zusammen. Verständlicherweise ergibt sich in PROLOG die deutlichste Abweichung von LISP, weil der Zugriff auf die Fakten der Datenbank nur über den Unifizierungsmechanismus (siehe Literaturliste 3) erfolgen kann. Die Prädikate nth/3 und cons/1 in Tab. 1c sind benutzerdefiniert und sollten dem jeweiligen Verwendungszweck angepaßt werden.

Die Assoziationsliste

Eine besondere Form von Liste ist in LISP die sogenannte Assoziationsliste (LOGO: Property Liste). Sie ermöglicht das Speichern von miteinander verbundenen Elementen. Weiter unten benutzen wir eine Assoziationsliste, um einen deutschsprachigen Simpeltext in ein holpriges Englisch zu übersetzen. Man speichert dann lediglich in einer Assoziationsliste die deutsch-englischen Assoziationen ab, also im unten aufgeführten Beispiel:

((dies this) (satz sentence) (ist is) (ein a))

Das erste Element einer Assoziationssubliste wird Schlüssel genannt (also: dies, satz, ist, ein). Verschiedene Listenfunktionen in LISP erlauben ein sehr effektives Arbeiten speziell mit

Funktion	Ausführung in LISP
Bindung der Liste (a, b, c) an das Symbol liste	(**setq** liste '(a, b, c)) oder (**setq** liste (**list** 'a 'b 'c))
Ausgabe des 1. Elementes **a**	(car liste)
Ausgabe der Restliste **(b, c)**	(**cdr** liste)
Ausgabe des 2. Elementes **b**	(**car** (**cdr** liste)) oder (**cadr** liste)
Ausgabe des n. Elementes	(**nth** n liste)
Einfügen eines neuen Elementes x in die Liste. Die neue Liste ist **(x, a, b, c)**	(**cons** x liste)
Aneinanderhängen zweier Listen. Ergibt **(a, b, c, d, e, f)**	(**append** '(a, b, c) '(d, e, f))

Tabelle 1a: Elementare Listenoperationen in LISP

dieser Listenform.

Bäume als spezielle Listen

Wir haben schon gesehen, daß die Elemente von Listen wiederum Listen sein können. Diese Datenstruktur nennt man besser einen Baum. Zur Veranschaulichung soll uns die folgende Liste dienen:

(k1 (k2 b1 b2) (k3 (k4 b3 b4) (k5 b5 (k6 b6 b7))))

Den zugehörigen Baum zeigt Abb. 1. Es bedeutenhierin k1 – k6 die Knoten des Baumes, b1 – b7 die Blätter des Baumes. Kenner werden einen sogenannten Binärbaum wiedererkennen; solche, die es werden wollen, lesen bei Literaturliste 4 nach. k1 ist hier die Wurzel des binären Beispielbaumes.

Funktionen in LISP und LOGO

Wir verlassen nun erst einmal den gemeinsamen Teil und kümmern uns speziell um die Organisation eines LISP- bzw. LOGO-Programmes. PROLOG ist eine relationale Sprache, in der das Programm aus Fakten und miteinander verknüpften Klauseln besteht, während LISP und LOGO als funktionale Sprachen ihr Ergebnis aus dem sequentiellen Aufruf von Funktionen erhalten. Wie in Pascal gibt es auch LISP- und LOGO-Programmteile, die zwar behandelt werden wie Funktionen, aber keinen Funktionswert liefern. Sie heißen (wie in Pascal auch) Prozeduren. Allgemein gilt, daß Prozeduren dann keine Funktionen sind, wenn sie Nebeneffekte zeigen. Selbstverständlich gibt es auch in LISP und LOGO standardmäßig vorgesehene Funktionen und Prozeduren; sie heißen Primitive. **setq** ist ein Beispiel für eine LISP-Prozedur, die keine Funktion ist, weil sie als Nebeneffekt einen Wert (2. Argument) an ein Symbol (1. Argument) bindet. Die Primitive **car** und **cdr** in Tab. 1a sind natürlich Funktionen, die ein entsprechendes Atom bzw. eine Liste als Funktionswert zurückgeben. Und da sie ihr Argument nicht verändern, haben sie auch keine Nebeneffekte. Tab. 2 zeigt eine Zusammenfassung aller Listenprimitive in XLISP mit Beispielen. Um Übersichtlichkeit zu wahren, werde ich hier lediglich die LISP-Schreibweise erwähnen. Anhand der Tab. 1 sollte der Leser in der Lage sein, die entsprechende LOGO-Schreibweise zu erarbeiten. Die schon in Tab. 1 er-

Funktion	Ausführung in LOGO
Bindung der Liste [A, B, C] an das Symbol LISTE	MAKE "LISTE [A, B, C]
Ausgabe des 1. Elementes A,	FIRST :LISTE
Ausgabe der Restliste [B, C]	BUTFIRST :LISTE
Ausgabe des n. Elementes	ITEM n :LISTE
Einfügen eines neuen Elementes X in die Liste. Die neue Liste ist [X, A, B, C]	MAKE "LISTE (FPUT "X, :LISTE)

Tabelle 1b: Elementare Listenoperationen in LOGO

Funktion	Ausführung in PROLOG
Nimmt das Fakt, das [a, b, c] eine Liste ist in die Datenbank auf.	assert(liste([a,b,c])).
Ausgabe des 1. Elementes der Liste in der Datenbank a	liste([Erstes|_]).
Ausgabe der Restliste [b, c]	liste([_|Rest]).
Ausgabe des 2. Elementes b	liste([_,Zweites|_]).
Ausgabe des n. Elementes	liste(X),nth(n,X,Element). mit: nth(1,[Kopf|Rumpf],Kopf). nth(N,[Kopf|Rumpf],X :- sum(N1,1,N),nth(N1,Rumpf,X).
Einfügen eines neuen Elementes x in die Liste in der Datenbasis. Die neue Liste ist [x, a, b, c]	cons(x). mit cons(X) :- liste(Y),assert(liste([X|Y])), RETRACT(liste(Y)).
Aneinanderfügen zweier Listen. Ergibt **Liste = [a, b, c, d, e, f]**	**append**([a, b, c],[d, e, f], Liste).

Tabelle 1c: Elementare Listenoperationen in PROLOG

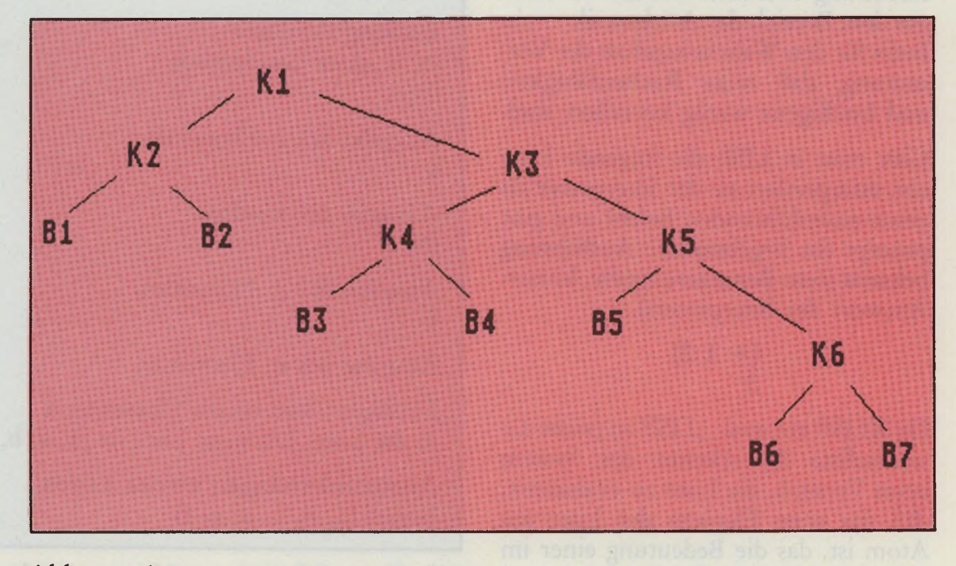
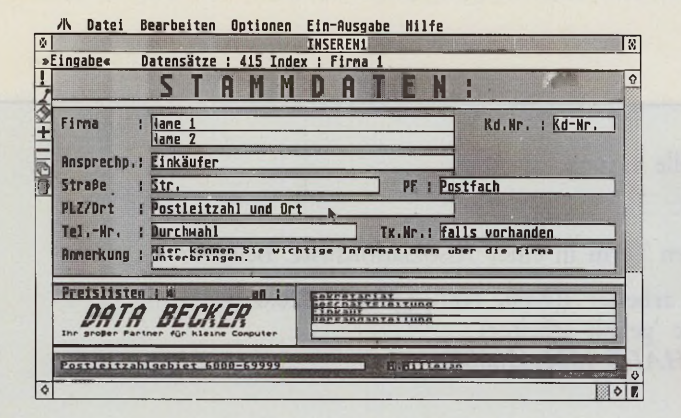


Abb. 1: Binärer Beispielbaum



Ihr bester Informant – DATAMAT ST

Keine Frage – Daten sind nur dann wertvoll, wenn sie immer auf dem aktuellsten Stand sind. Wenn sie also entsprechend verwaltet und gepflegt werden. Eine wirklich leistungsfähige Dateiverwaltung kann den damit verbundenen Zeitaufwand auf ein Minimum beschränken und sucht Ihnen auch schnell und zuverlässig die gewünschte Information. Nur welche Dateiverwaltung ist die richtige? Nun, einige sind besonders schnell, aber dafür meist kompliziert zu bedienen. Andere hingegen sind sehr benutzerfreundlich, aber oft zu langsam. DATAMAT ST glänzt mit hoher Geschwindigkeit beim Blättern, Suchen und Editieren und bleibt dabei doch durch die komplette Maussteuerung unter GEM extrem leicht zu bedienen. Jederzeit, selbst während eines Telefongesprächs, können Sie so die nötigen Informationen abrufen. DATAMAT ST – einen besseren Informanten können Sie kaum finden.

DATAMAT ST in Stichworten: Voll GEM-orientiert – komplett mausgesteuert – Bildschirmmasken mit vielen Grafikelementen – Textdarstellung in verschiedenen Größen und Stilen – Maskeneditierung während der Dateibearbeitung möglich – 4 Dateien gleichzeitig im Zugriff – max. 64000 Datensätze pro Datei, max. Datensatzlänge 64 KByte – Schnittstelle zu TEXTOMAT ST und BECKERtext ST für Serienbriefe – Anpassung an alle gängigen Drucker – schnelle, komfortable Such- und Sortierverfahren – läuft auf Monochrom- und Farbbildschirm – mit ausführlichem, deutschem Handbuch.

DATAMAT ST DM 99,–

bekommt Unterstützung: DATAMAT-ST-Anwendungen

Neben der erfolgreichen Dateiverwaltung DATAMAT ST entwickelten die Programmierer DATAMAT-ST-Anwendungen. Ein viel zu bescheidener Name für dieses Produkt. Denn DATAMAT-ST-Anwendungen ist keine bloße Sammlung hilfreicher Anwendungen zu DATAMAT ST. Hier finden Sie ebenso eine Vielzahl nützlicher Hilfsprogramme und Erweiterungen, die Sie zum Teil auch vollkommen losgelöst von DATAMAT ST verwenden können: Spoolerprogramme zum Beschleunigen der Druckerausgabe, ein Hilfsprogramm zum Einlesen von Disketteninhaltsverzeichnissen, zur Mehrfachverwendung von Datenfeldern in TEXTOMAT ST und BECKERtext ST oder auch Tastaturtreiber zur Darstellung fast aller Zeichen des ATARI Zeichensatzes – um nur einige zu nennen. Die eigentlichen Anwendungen sind natürlich flexibel und lassen sich beliebig nach Ihren Wünschen verändern und ausbauen.
Die Anwendungen im einzelnen: Adreßdatei mit Serienbrieferstellung – Briefdatei zur Erstellung von Standardbriefen – Dateisystem zur Kunden-/Lagerverwaltung mit der Möglichkeit, Rechnungen und Mahnungen zu erstellen – Diskettendatei zur Verwaltung von Disketteninhalten (z.B. in Verbindung mit dem mitgelieferten Accessory) – Etikettendatei zur Beschriftung von Aufklebe-Etiketten – Kassettendatei zur Verwaltung von Musikkassetten mit der Möglichkeit, Kasetteneinleger zu drucken – Literaturdatei zur Verwaltung von Literaturverzeichnissen – Schachdatei zur grafischen Darstellung und Archivierung von Schachstellungen – Vokabeldatei zum Verwalten oder Erlernen einer Fremdsprache – Rezeptdatei zur Verwaltung von Kochrezepten o.ä. – Notizblock, Terminkalender.

DATAMAT-ST-Anwendungen DM 99,–

DATA BECKER
Merowingerstr. 30 · 4000 Düsseldorf · Tel. (0211) 310010

BESTELL-COUPON
Einsenden an: DATA BECKER · Merowingerstr. 30 · 4000 Düsseldorf 1
Bitte senden Sie mir:
☐ DATAMAT ST DM 99,–
☐ DATAMAT ST Anwendungen DM 99,–
zzgl. DM 5,– Versandkosten
☐ per Nachnahme ☐ Verrechnungsscheck liegt bei
Name
Straße
Ort

wähnten Primitive **car, cdr** und **cons** bilden so etwas wie ein Funktions-Umkehrfunktionspaar. D. h., es gilt die Identität:

(cons (car liste) (cdr liste)) ≡ liste

Mit **cons** wird also eine durch **car** und **cdr** aufgetrennte Liste wieder zusammengefügt. Bitte erinnern Sie sich, daß in einer Liste nicht nur Atome, sondern auch Listen als Elemente stehen können. So ist z. B. ((a b) (c d)) eine Liste, die aus zwei Elementen besteht: der Liste (a b) und der Liste (c d). Folglich ergibt (car '((a b) (c d))) den Funktionswert (a b), eine Liste als erstes Element. (cdr '((a b) (c d))) liefert dann logischerweise ((c d)), die Restliste, die die Liste (c d) als einziges Element enthält. Für die in Tabelle 2 aufgeführten Funktionen gibt es noch Optionen, deren Besprechung den Rahmen dieser Einführung sprengen würde. Der Leser sei hier auf das XLISP-Manual verwiesen.

Evaluierung von LISP-Termen

Erfahrungsgemäß bereitet Anfängern die Evaluierung von LISP-Termen große Schwierigkeiten. Deshalb möchte ich die in Literaturliste 2 recht kurz geratene Einführung hier ergänzen. Generell versucht LISP (genauer gesagt ein **eval**-Lisp), den Wert eines Terms zu erfahren, bevor es ihn weiterverarbeitet. Zur Anschauung betrachte man den XLISP-Dialog in Abb. 2. In der ersten Zeile wird das Atom A an den Wert 1 gebunden. Da ein Atom (eine Zahl ist ein numerisches Atom) zu sich selbst evaluiert, ist in der Anweisung kein Apostroph für 1 erforderlich. Eigentlich müßte das Symbol A mit einem Apostroph eingegeben werden, um dem Interpreter zu verdeutlichen, daß das Symbol A als unevaluiertes Atom verwendet werden soll. Dies ist nicht erforderlich, weil meist eine Bindung ohnehin nur an ein Atom erfolgt und LISP deshalb vorsorglich das erste Argument mit einem Apostroph versieht. **setq** ist scchließlich nichts anders als die Kurzform von **set quote**. (setq a 1) ist also äquivalent zu (set 'a 1). In der zweiten Anweisung wird schließlich das Atom B an die Liste (+ a 1) gebunden. Dies ist zwar die Anweisung, die Summe von a und 1 zu berechnen, da sie aber in der **setq** Anweisung quotiert wird, wird sie nicht ausgeführt. Wachsame haben vielleicht bemerkt, daß hier be-

append
Siehe Tabelle 1a.

assoc
Findet einen Term in einer Assoziationsliste. Beispiel:

(setq arbeiter '((name emil) (gehalt 1526.65)))
(assoc 'gehalt arbeiter)
(GEHALT 1526.650000)

car
Siehe Tabelle 1a.

cdr
Siehe Tabelle 1a.

cxxr
xx bedeutet eine beliebige Kombination von a und d. z. B. cadr oder cdar oder cddr oder caar. Siehe Tabelle 1a.

cxxxr
Siehe cxxr und Tabelle 1a.

cxxxxr
Siehe cxxr und Tabelle 1a.

cons
Siehe Tabelle 1a.

last
Ergibt die Liste, welches das letzte Element enthält. Beispiel:

(last '(a b c))
(c)

length
Gibt als Funktionswert die Anzahl der in der Liste enthaltenen Elemente an. Beispiel:

(length '(a b (c d)))
3

list
Der Funktionswert ist eine Liste der Argumente. Beispiel:

(list 'a 'b '(c d))
(a b (c d))

mapc
Wendet die Funktion (1. Argument) nacheinander auf die Listen der Argumentwerte (2. – n. Argument, eine Liste je Argument der Funktion) an. Als Funktionwert wird die erste Liste von Argumenten ausgegeben (2. Argument von mapc). Beispiel:

(mapc + '(1 2 3) '(4 5 6)))
(1 2 3)

Man beachte, daß von der Summation nichts bemerkt wird, da die Funktion + (Addition) nebeneffektfrei ist und die Ergebnisse einfach vergessen werden. Diese Listenfunktion sollte deshalb nur bei Funktionen mit Nebeneffekt angewendet werden. Beispiel:

(mapc (lambda (x y) (prin1 (+ x y))) '(1 2 3) '(4 5 6))
579(1 2 3)

Über die Bedeutung von **lambda** später.

mapcar
Wie **mapc**, allerdings wird hier als Funktionswert die Liste der Funktionswerte zurückgegeben. Beispiel:

(mapcar + '(1 2 3) '(4 5 6))
(5 7 9)

mapl
Wie **mapc**, wendet die Funktion aber auf die **cdr** der Argumentlisten an. Beispiel:

(mapl length '(a b c))
(a b c)

Man beachte auch hier, daß die Funktion Nebeneffekte haben muß, um eine Wirkung zu zeigen.

maplist
Wie **mapcar**, wendet die Funktion aber au die **cdr** der Argumentlisten an. Beispiel:

(maplist length '(a b c))
(3 2 1)

Zuerst ist die Argumentliste (a b c) und deren Länge 3. Der **cdr** dieser Liste ist (b c), die Länge 2. Der **cdr** dieser Liste ist (c), die Länge 1. Dann bricht **maplist** ab, weil der folgende cdr NIL ist.

member
Findet einen Ausdruck in einer Liste und gibt die Restliste zurück. Beispiel:

(member 'b '(a b c))
(b c)

nth
Siehe Tabelle 1a.

nthcdr
Gibt den n-ten **cdr** einer Liste zurück. N=0 ergibt die Originalliste. Beispiel:

(nthcdr 2 '(a b c d e f))
(c d e f)

remove
Entfernt einen Term aus einer Liste. Beispiel:

(remove 'b '(a b c))
(a c)

reverse
Kehrt eine Liste in der Reihenfolge ihrer Elemente um. Beispiel:

(reverse '(a b c))
(c b a)

sublis
Ersetze mit Hilfe einer Assoziationsliste. Beispiel:

(sublis '((halli hallo)) '(aber halli))
(ABER (HALLO))

subst
Ersetzt LISP Terme. Beispiel:

(subst 'demo 'b '(a b c))
(a demo c)

Tabelle 2a: Listenfunktionen in XLISP

reits die in Pascal, C, Basic & Co. streng durchgeführte Trennung von Programm und Daten verwischt. Hier dient eine **Programmanweisung** (addiere den Wert von a und 1) als **Wert**. Läßt man die Quotierung für das zweite Argument weg, dann wird die Evaluierung des zweiten Terms durchgeführt. D. h., die Summe von A und 1 wird erst berechnet und dann das Ergebnis an C gebunden (dritte Anweisung). Das Ergebnis der virten Anweisung zu verstehen, dürfte nun keine Schwierigkeiten bereiten: A evaluiert zu 1, C evaluiert zu 2 und die Summe ist 3. Wer nun in der fünften Anweisung (+ a b) ein vernünftiges Ergebnis erwartet, hat zu hohe Erwartungen an den XLISP-Interpreter. Selbstverständlich evaluiert XLISP den Term zu (+ 1 (+ A 1)), aber die Evaluierung wird nicht rekursiv bis zur letzten Stufe durchgeführt, sondern stoppt hier. Da die Addition zwei numerische Atome verknüpfen will, (+ A 1) aber eine Liste darstellt, ist nun die Fehlermeldung **bad argument type** zu erwarten. Sorgt man mit **eval b** jedoch ausdrücklich für eine Evaluierung des Termes B, dann wird in der Tat die korrekte Summe ermittelt (sechste Anweisung):

(+ a (eval b)) → (+ a (+ a 1)) →
(+ 1 2) → 3

Einige einfache Anwendungen

Ungeachtet der Tatsache, daß LISP keine Sprache ist, in der man bevorzugt rechenintensive Vorgänge bearbeitet, betrachten wir als einfachstes Beispiel die Berechnung einer Wertetabelle. Dies läßt sich folgendermaßen erreichen:

(setq argumente '(0.0 0.785398 1.04719
1.57079 3.14159))
(setq wertetabelle (mapcar sin
argumente))

Diese zwei Zeilen berechnen den Sinus von $0, \pi/4, \pi/3, \pi/2$ und π. Die Funktionswerte findet man anschließend in der Wertetabelle wieder:

wertetabelle
*(0.000000 0.707106 0.866021
1.000000 0.000020)*

Ein weiteres einfaches Beispiel unter Verwendung der Listenfunktionen ist die Übersetzung eines deutschsprachigen Texten in *pitching english*. D. h., man übersetzt den Text wortweise und erhält ein mehr oder weniger

brauchbares Konglomerat englischer Worte, die im Idealfall einen lesbaren Satz bilden. Dazu stellt man sich zunächst ein Lexikon in Form einer Assoziationsliste zusammen, deren erste Element jeweils das deutsche Wort und dessen zweites Element das zugehörige englische Wort darstellen. Dann läßt man die Assoziationsliste die entsprechenden Substitutionen durchführen und entfernt anschließend die störenden Klammern aus dem Text:

(setq deutsch__englisch '((dies this)
(satz sentence) (ein a) (ist is))
((DIES THIS) (SATZ SENTENCE)
(EIN A) (IST IS))
(setq deutsch '(dies ist ein satz))
(DIES IST EIN SATZ)
(setq englisch (sublis deutsch__englisch
deutsch))
((THIS) (IS) (A) (SENTENCE))
(setq englisch (mapcar car englisch))
(THIS IS A SENTENCE)

Die letzten beiden Zeilen lassen sich natürlich noch zu einer zusammenfassen:

...
(DIES IST EIN SATZ)
(setq englisch (mapcar car(sublis
deutsch__englisch deutsch)))
(THIS IS A SENTENCE)

Die eigentlich interessanten Anwendungen erhält man allerdings erst, wenn man seine eigenen Funktionen definieren kann. Dazu kommen wir aber erst in der nächsten Folge. Dann nämlich geht es um die Definition von Funktionen und das LAMBDA-Konzept. Wie Sie sehen, bleiben wir also noch ein wenig im LISP/LOGO-Sprachraum. Erst wenn wir die Prädikatsfunktionen besprechen, werden wir unseren Blick wieder stärker in Richtung PROLOG Lenken. Schließlich ist PROLOG die Sprache zur Verarbeitung logischer Prädikate.

Karl Sarnow

[1] Sarnow, K. Einführung in die künstliche Intelligenz. ST-Computer 11/86.

[2] Sarnow, K. XLISP Review. ST-Computer 1/87

[3] Sarnow, K. TOY-PROLOG Review. ST-Computer 2/87.

[4] Wirth, Niklaus. Algorithmen und Datenstrukturen. B. G. Teubner, 1975.

delete
Löscht einen Term aus einer Liste. Beispiel:

(setq liste '(a b c))
(A B C)
(delete liste 'b)
(A C)

nonc
Vereinigt zwei Listen zu einer (physisch). Beispiel:

(setq liste1 '(a b c))
(A B C)
(setq liste2 '(d e f))
(D E F)
(nonc liste1 liste2)
(A B C D E F)
liste 1
(A B C D E F)
liste 2
(D E F)

rplaca
Ersetzt den car einer Liste. Beispiel:

(setq liste '(a b c))
(A B C)
(rplaca liste 'b)
(B B C)
liste
(B B C)

rplacd
Ersetzt den cdr einer Liste. Beispiel:

(setq liste '(a b c))
(A B C)
(rplacd liste '(d e))
(A D E)
liste
(A D E)

Tabelle 2b: Listenprozeduren in XLISP (destruktiv)

> (setq a 1) <------	A wird an den Wert 1 gebunden
1	
> (setq b '(+ a 1)) <------	B wird an die Liste (+ A 1) gebunden
(+ A 1)	
> (setq c (+ a 1)) <------	C wird an den Wert der evaluierten
2	Liste (+ A 1) gebunden, d. h. an 2.
> (+ a c) >----------	A evaluiert zu 1, C evaluiert zu 2,
3	die Summe beider ergibt 3.
> (+ a b)	
error: bad argument type – (+ A 1)	B evaluiert zu (+ A 1), das ist der
1:>	falsche Argumenttyp für eine Addition.
> (+ a (eval b)) >-------	Eval b evaluiert b zu (+ A 1), dies wird
3	zu 2 evaluiert, A wird zu 1 evaluiert,
>	die Summe ergibt 3.

Abb. 2: Protokoll eines XLISP Dialoges zum Problem der Evaluierung



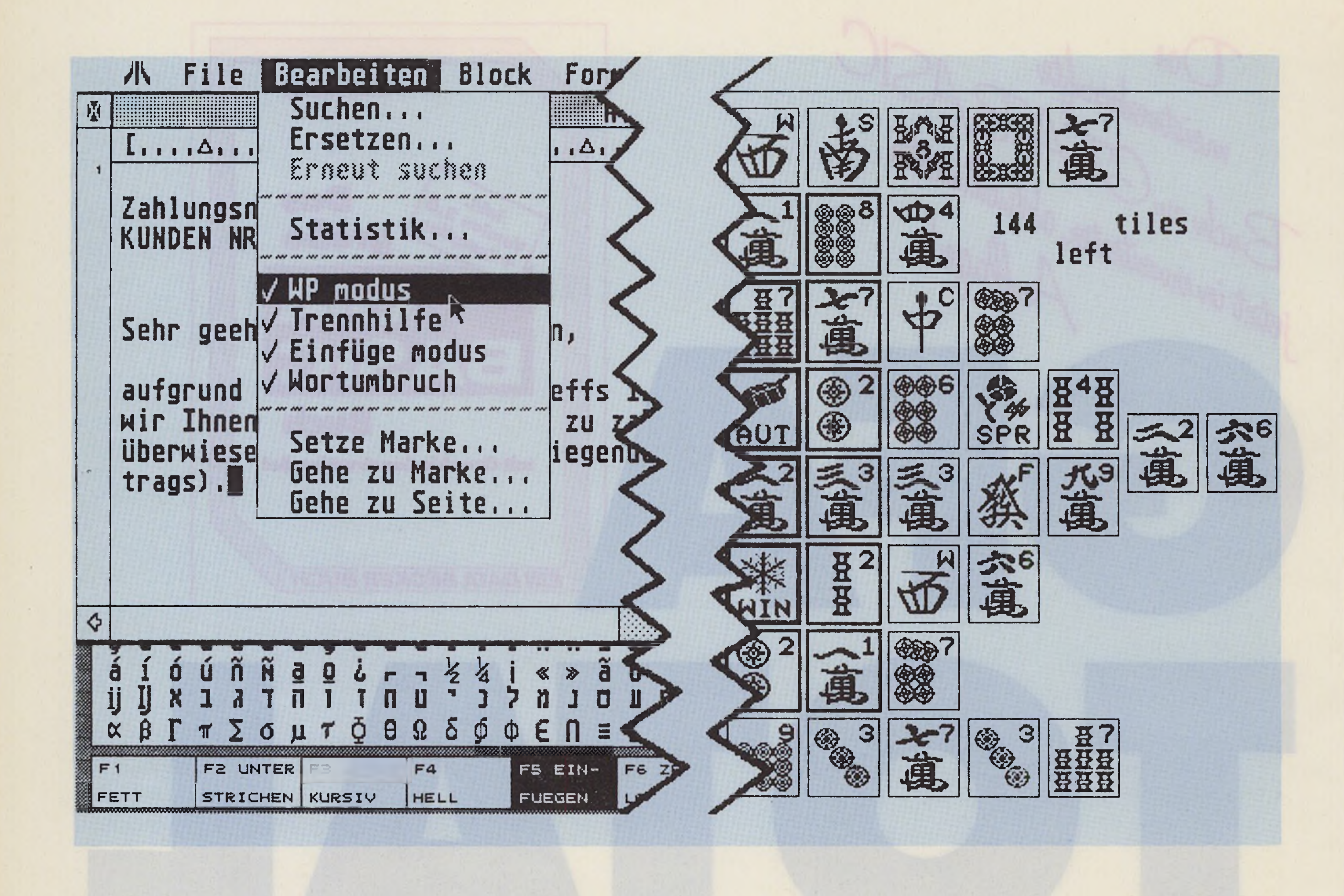
GFA TOTAL

Die Idee zum Buch: Anhand eines kompletten, ausbaufähigen Programms sollte der mächtige Befehlssatz des GfA-BASICs vermittelt und erklärt werden. Eine Idee, die den Autor so begeisterte, daß er ein Grafik-Programm der Superlative entwickelte. Im Untertitel bescheiden „Graphic Construction Set" genannt, bietet es alle Standards und zusätzlich zahlreiche, weitere Funktionen: Trickfilm und Animation in Echtzeit, Icon- und Pattern-Editor, Trommelgenerator, 3-D-Animation in atemberaubender Geschwindigkeit, Clipping, Patch-Grafiken... Die Möglichkeiten, die Uwe Litzkendorf hier zeigt, sind fast schon unglaublich. Aber das wichtigste: Nachdem man das Buch gelesen hat, weiß man, wie man sowas programmiert; kennt man alle GfA-Befehle. Denn beim Erstellen des Programms wird jeder neue Befehl detailliert erklärt – und zwar genau dann, wenn er gebraucht wird. Immer eingebunden in kleine Prozeduren, die wiederum in das Programm integriert, aber auch für eigene Programme verwendet werden können. Hierbei lernen Sie dann auch die zahlreichen Tips und Tricks, wie z. B. Rastertechnik, Sprites, System Calls, GEM-Prozeduren, Window- und Objektprogrammierung. Selbstverständlich, daß in diesem Buch auch eine komplette Befehlsübersicht zu finden ist. Bei solch kompaktem Detailwissen konnte der Erfolg dieses Buches nicht überraschen. Doch dann kam GfA-BASIC 2.0 – das große GfA-BASIC-Buch mußte aktualisiert werden. Kurzerhand ergänzte der Autor das Buch um die neuen Befehle der Version 2.0 und fügte auch gleich dem Programm selbst noch zahlreiche Features hinzu. Ebenfalls in der jetzt erweiterten Auflage: Eine ausführliche Beschreibung des GfA-Compilers und für alle, die einen wirklich einfachen Einstieg suchen, ein detaillierter Einsteigerteil.

Das große GfA-BASIC-Buch
Hardcover
ca. 600 Seiten
DM 49,–

DATA BECKER
Merowingerstr. 30 · 4000 Düsseldorf · Tel. (0211) 31 00 10

BESTELL-COUPON
Einsenden an: DATA BECKER · Merowingerstr. 30 · 4000 Düsseldorf 1
Bitte senden Sie mir:
zzgl. DM 5,- Versandkosten
☐ per Nachnahme ☐ Verrechnungsscheck liegt bei
Name
Straße
Ort



Geteilte Freude ist doppelte Freude...

...jedenfalls, wenn man genug davon hat.

Kumas K-Switch im Test

Es ist eine weise Erkenntnis, die uns beim Test des K-Switch in den Sinn kam. Denn verdoppelte es nicht wirklich die Freude jedes Atari-Benutzers, sein geliebtes Gerät zu teilen und zwei seiner Lieblingsprogramme auf Knopfdruck zur Verfügung zu haben – nicht das Desktop, sondern gleich zwei, und das alles auf einem Gerät? Und wäre es nicht eine Freude für jedes Programm, nicht länger alleine die verschlungenen Pfade der Rechnerarchitektur zu durcheilen, sondern sich in trauter Zweisamkeit durchs Leben zu schlagen?

Zur Teilung Ihres Rechners müssen Sie nicht länger zu Axt oder Säge greifen. Ein kleines Accessory mit dem unscheinbaren Namen K-Switch nimmt Ihnen diese Mühe ab. Von der etwas sanfteren Art der Teilung, die Ihnen K-Switch bietet, haben Sie einige Vorteile.

Was tut das Accessory denn nun genau? K-Switch teilt den Speicher Ihres Rechners in zwei Hälften. In jedem dieser so eingerichteten Halb-Rechner kann sich nun ein eigenes Programm häuslich niederlassen. Um Mißverständnissen vorzubeugen: Mit Multitasking oder ähnlichem hat K-Switch nichts zu tun. Die beiden Programme laufen nicht gleichzeitig oder quasigleichzeitig, sondern immer nur abwechselnd, vom Benutzer auf Knopfdruck aktivierbar. Leider wird sich die Freude an Ihrem ST nur dann verdoppeln, wenn beide Programme sich mit dem jeweils halben Speicher zufrieden geben. 874 Kilobyte stehen insgesamt zur Verfügung, wenn alle anderen Accessories verschwunden sind. Für jede Speicherhälfte bleiben also noch 437 Kilobyte freier Speicher. Falls Sie noch

eine oder zwei Ram-Disks wollen, wird es für viele Programme schon recht eng. Aber dazu später mehr.

Wie geht das ST-Zersäge-Kunststück nun vor sich? Nach dem Booten der K-Switch-Disk (oder besser einer Kopie, das Programm ist nicht kopiergeschützt), befindet sich das Accessory in der Menüleiste. Wenn sich auf der Disk noch andere Accessorys befinden, sind auch sie an ihrem gewohnten Platz in der Menüleiste zu finden. Auto-Ordner können mit K-Switch nicht verwendet werden, sie entfernen das Programm nach dem Booten gleich wieder.

Ruft man den K-Switch Menüeintrag auf, erscheint eine Dialogbox, die folgende Möglichkeiten bietet:

Installation:
Hiermit wird der Switcher installiert, wobei alle Optionseinstellungen (siehe unten) übernommen werden.

Dateizugang:
Hier finden sich besonders angenehme Funktionen, die es z. B. ermöglichen, eine komplette Speicherhälfte auf Disk zu speichern und wieder zu laden. Damit kann man sich gegen Abstürze in Testphasen absichern.

Optionen:
Unter dieser Bezeichnung versteckt sich z. B. eine besondere, in der Größe einstellbare Ram-Disk, auf die beide Speicherhälften zugreifen können und die damit die schnellste Transfer-Möglichkeit zwischen zwei Programmen unter K-Switch bietet. Optionseinstellungen können abgespeichert werden und stehen dann beim nächsten Booten automatisch zur Verfügung.

Werden keine anderen Optionen eingestellt, installiert das Programm automatisch eine 150 Kilobyte große Ram-Disk, die für beide Hälften zugänglich ist, falls nicht vorher eine andere Ramdisk eingerichtet wurde, z. B. mit G-Ram oder K-Ram oder einem anderen Ram-Disk-Programm. Im letzteren Fall kann die erste Speicherhälfte nur mit der selbst eingerichteten Pseudo-Floppy verwendet werden, die zweite Speicherhälfte greift dafür alleine auf die von K-Switch installierte Ram-Disk zurück. Versucht man, eine weitere Ram-Disk, egal mit welchem Laufwerkskennbuchstaben, **nachträglich** einzurichten, passiert genau das gleiche: Die von K-Switch installierte ist aus der ersten Hälfte nicht mehr zu erreichen. Wird die neu eingerichtete Ram-Disk jedoch wieder abgeschaltet, ist die K-Switch Ram-Disk sofort von beiden Seiten aus zugänglich (sogar ohne Zerstörung ihres Inhalts), vorausgesetzt, Sie benutzen unterschiedliche Laufwerkskennungen.

Nach der Installation von K-Switch (die übrigens nicht sofort nach dem Booten, allerdings unbedingt auf dem Desktop erfolgen muß) kann durch gleichzeitiges Drücken beider Shift-Tasten und der Alternate-Taste zwischen den beiden Hälften umgschaltet werden.

Die zweite Hälfte präsentiert uns dann das Desktop, auf dem sogar die Accessories getrennt von denen des anderen Desktops funktionieren. So wirken sich z. B. Kontrollfeld- oder Druckereinstellungen der einen Seite überhaupt nicht auf der anderen aus, auch ein Taschenrechnergedächtnis auf einer Seite unterscheidet sich von dem der anderen. Man lebt also nebeneinander her und kommuniziert nur noch über Ram-Disks.

Eher eine zerrüttete Ehe als trautes Zusammensein, aber sehr nützlich: Man findet sein Programm nämlich immer in genau dem Zustand wieder, in dem man es verlassen hat. Man kann sogar komplette Speicherhälften oder auch nur die gemeinsame Ram-Disk en bloc auf Disk speichern. Nur eines sollte man beachten:

Unterbrechen Sie nie laufende Disk-, Drucker- oder sonstige Peripherie-Operationen. Auch interruptgesteuerte Programme, zum Beispiel Sequencer, erzeugen bei Unterbrechungen manchmal skurrile Effekte.

Alle von mir in Zusammenarbeit mit K-Switch getesteten Programme liefen reibungslos, so z. B. 1_st Word, K-Graph, das Kuma Resource Construction Set, CCD Pascal, Megamax C, Borrowed Time und Squixx. Der MacIntosh-Emulator von Robtek war, wie nicht anders zu erwarten, nach Installation von K-Switch nicht lauffähig. Das wäre ja auch zu schön, um wahr zu sein: Auf Knopfdruck umschalten von Atari auf Apple... Auch Grafikprogramme wie Degas Elite benötigen zuviel Speicherplatz und laufen deshalb nicht.

Um noch einmal zusammenzufassen: Soweit ich es beurteilen kann, laufen alle Anwendungen, die

1.) keinen Auto-Ordner benötigen und
2.) die sich mit dem verbleibenden Speicherbereich zufriedengeben,

problemlos mit K-Switch zusammen. Da aber die Atari-Anwendungen immer speicherplatzintensiver werden, wird sich das Programm wohl erst auf den 2- und 4-Megabyte Ataris richtig bewähren. Im Moment ziehe ich noch die Arbeit mit einer großen Ram-Disk vor; das Nachladen geht im Zweifelsfall fast genauso schnell. Aber wenn der größere Speicher erst da ist... Wer jedoch Anwendungen, hat, die mit dem eingeschränkten Speicher auskommen, dem sei K-Switch wärmstens empfohlen.

Noch ein Wort zur Ausstattung: Die Dialoge des Programms sind in Deutsch abgefaßt, ebenso die (ausreichend ausführliche) Anleitung.

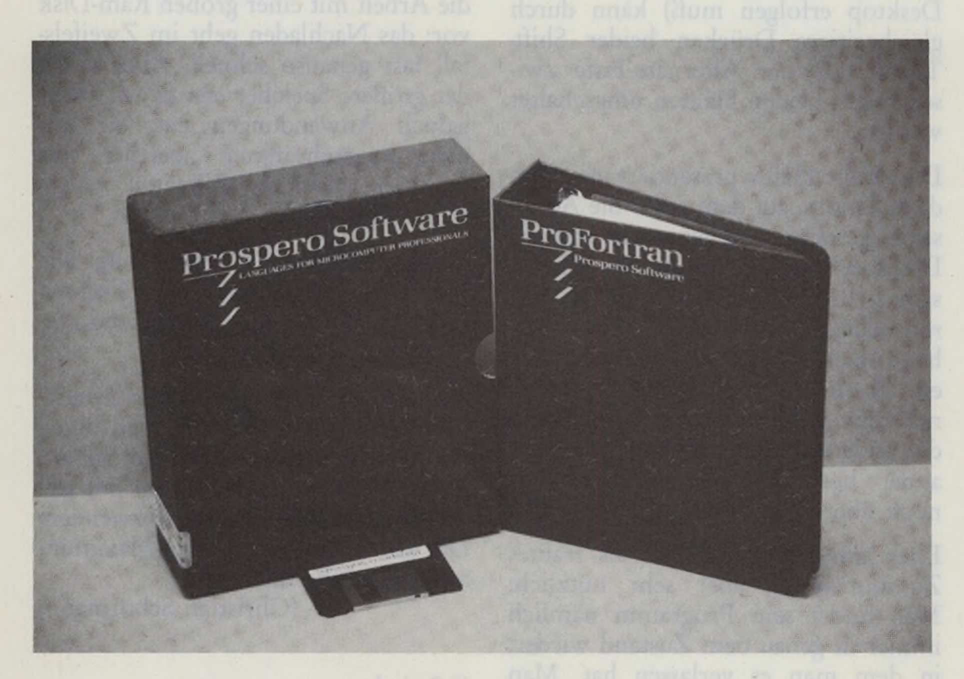
Für die kommende Speicher-Generation kann man ja vielleicht auch auf einen Update hoffen, der es ermöglicht, mehr als zwei Anwendungen mit frei zuteilbaren Speicher zu verwenden. Das wäre dann wirklich ein Maximum an geteilter Freude.

(Christian Schormann)

K-Switch
Von Kuma Computers Ltd.
Diverse Händler
Preis: ca. 98,– DM

ProFortran 77

– Die feine englische Art



Für den Atari ST gibt es inzwischen nicht nur eine Fülle äußerst nützlicher Programme – man denke etwa an Textverarbeitung, Zeichnen, Desktop-Publishing, Dateiverwaltung, Tabellenkalkulation, Branchenpakete und vieles mehr – sondern auch eine große Auswahl an Programmiersprachen. Neben Einsteigern und Gelegenheitsprogrammierern, die mit Basic und Logo schon in der Grundausstattung bedient werden, finden auch viele Profis die richtige Sprache. Dabei ist Vielfalt gefragt: Es gibt noch keine Programmiersprache, die sich für alle Zwecke gleichermaßen eignet.

Wenn ein Anwender auch mit einem Großrechner arbeitet, dann möchte er natürlich auf seinem Mikrocomputer nicht nur die gleiche Sprache, sondern auch den gleichen Sprachumfang vorfinden. Andererseits muß natürlich auch das Preisniveau dem privaten Budget entsprechen.

Preiswert und gut

ProFortran ist ein Fortran-System, das von der englischen Firma Prospero Software erstellt wurde und in Deutschland für knapp 500 Mark verkauft wird. Dieselbe Firma bietet auch ProPascal an, das bereits im Heft 10/86 dieser Zeitschrift besprochen wurde. Was von dem Streit um die bessere Programmiersprache zu halten ist, wird an folgender Tatsache deutlich: Beide Systeme, also ProPascal und ProFortran, sind mit ProPascal erstellt und verwenden das gleiche Laufzeitsystem. Jede Sprache hat ihr spezielles Einsatzgebiet.

Das ProFortran-Paket besteht aus einer einseitig formatierten Diskette und einem Handbuch. Die Diskette ist nicht kopiergeschützt; das bei anderen Produkten oft notwendige Einlegen der Originaldiskette beim Programmstart entfällt. Im Hinblick auf den günstigen Preis, den Umfang der mitgelieferten Dokumentation und die Möglichkeit, gegen Einsendung von 30 Mark und Originaldiskette ein Update zu bekommen, sollte dem Vertreiber dadurch kein Schaden entstehen. Professionelle Anwender sind wegen der Dokumentation und der Update-Möglichkeit ohnehin auf die Originalversion angewiesen, die Benutzer von Kopien sind gewiß keine ernsthaften Daueranwender und somit auch keine verlorenen potentiellen Käufer.

Solide Dokumentation

Laut Handbuch läuft ProFortran auf jedem verfügbaren Atari der ST-Serie; das gesamte Programmpaket hat, einschließlich einiger Beispielprogramme, auf einer einseitig beschriebenen Diskette Platz. Hierzu muß man wissen, daß ProFortran auch für den Sinclair QL, dessen Prozessor 68008 ja aus der gleichen Familie stammt, angeboten wird. Dort sind die Speicherkapazitäten nicht so üppig wie auf dem Atari.

Die Dokumentation ist sehr solide und entspricht dem für Software zum IBM-PC üblichen Standard. Die Druckqualität ist gut, die Texte sind übersichtlich formatiert und gut lesbar. Es handelt sich um die von Fachleuten in englischer Sprache abgefaßte Originaldokumentation, was der erfahrene Fortran-Programmierer sicher zu schätzen weiß. Für Anfänger genügt sie natürlich nicht, hier gibt es jedoch gute und preiswerte Lehrbücher in deutscher Sprache (z. B. Müller, Streker-Seeborg: Fortran 77 Programmierungsanleitung, Bibliographisches Institut Mannheim, 1984).

Der erste Teil der Dokumentation gibt auf elf Seiten einen Überblick über wichtige Details der Implementierung, etwa die verfügbaren Datentypen, Hinweise zur Ein-/Ausgabe und ähnliches.

Anbindung an TOS und GEM

Der zweite Teil des Handbuchs entspricht etwa der „Programmer Reference“ in englischsprachiger Literatur zu Großrechnern. Auf 125 Seiten wird die Sprache vollständig und übersicht-

lich dargestellt, man kann in allen Zweifelsfällen nachschlagen und detaillierte Auskunft erhalten. Kapitel 8 behandelt „implementation-dependent aspects". Unter anderem wird hier die Einbindung von TOS-, VDI- und AES-Routinen gezeigt, für die eine Fülle von Interface-Programmen in den mitgelieferten Systembibliotheken zur Verfügung steht. Die Verknüpfung mit Assembler-Programmen wird ausführlich anhand von Beispielen erläutert.

Der dritte Teil heißt „ProFortran-77 Operation" und entspricht dem „user guide" auf Großrechnern. Er umfaßt, einschließlich der Anhänge, 58 Seiten. Hier wird der Umgang mit dem Programmpaket unter GEM und unter dem Befehlszeileninterpreter COMMAND.TOS beschrieben. Dabei wird die gemeinsame Entwicklung von ProFortran und ProPascal besonders deutlich: Das Handbuch von ProPascal ist genauso aufgebaut, die Texte sind sogar weitgehend identisch.

Gemeinsam mit Pascal...

Für den Programmierer haben diese Gemeinsamkeiten eine erfreuliche Konsequenz: Fortran-Programmeinheiten und Pascal-Module können in einem Programm gemischt werden. Anhang E des jeweiligen Handbuchs gibt hierüber Auskunft. Die einzelnen Programmteile werden mit ProFortran oder ProPascal getrennt übersetzt, beim anschließenden Binden ignoriert man einfach die unterschiedliche Herkunft. Dies ist möglich, weil beide Programmpakete den gleichen Binder (von GST), die gleiche Laufzeit-Bibliothek (s. u.) und die gleiche Datendarstellung verwenden. Viele der Laufzeit-Fehlermeldungen, die jeweils durch einen Buchstaben charakterisiert sind, stimmen in beiden Sprachen überein; Fehlermeldungen, die nur für eine der Sprachen typisch sind, werden über zusätzliche Buchstaben realisiert.

...und Assembler

Die Integration geht noch weiter: Neben Fortran- und Pascal-Programmen können auch mit dem Makro-Assembler von GST erzeugte Binärdateien verarbeitet werden. Dieser leistungsfähige Assembler mit der genauen Bezeichnung GST-ASM V1.0 wurde bereits im Heft 11/86 dieser Zeitschrift getestet.

Die Möglichkeit der Integration von Assembler-Teilen in Fortran-Programme macht ProFortran endgültig zum professionellen Werkzeug.

Sprachumfang: Komplettes Fortran 77

Der Sprachumfang von ProFortran umfaßt den gesamten Fortran-77-Standard gemäß DIN 66027 (identisch mit ISO 1539-1980 und der ANSI-Norm X3.9-1978). Insbesondere die geforderten Datentypen INTEGER, REAL, DOUBLE PRECISION, COMPLEX, LOGICAL, CHARACTER sind allesamt implementiert. Gemäß dem Fortran-Standard belegen INTEGER und REAL eine numerische Speichereinheit (dies sind hier 4 Byte, also 32 Bit oder ein Langwort), DOUBLE PRECISION und COMPLEX dagegen zwei. Als Größen vom Typ INTEGER sind damit ganze Zahlen zwischen −2147483647 und +2147483647 erlaubt; durch den Typ REAL lassen sich reelle Zahlen in der Größenordnung zwischen E−38 und E+38 und mit einer Genauigkeit von etwa 7 Dezimalstellen darstellen, bei DOUBLE PRECISION liegt die Größenordnung zwischen E−308 und E+308 und die Genauigkeit bei 16 Dezimalstellen. Der Typ CHARACTER verwendet ein Byte pro Zeichen, LOGICAL belegt für jeden Wert eine numerische Speichereinheit. Zusätzlich zum Standard gibt es noch die Datentypen INTEGER★2, LOGICAL★2 mit 2 Byte Speicherplatz für jede Größe und INTEGER★1, LOGICAL★1, die jeweils nur 1 Byte belegen. Entsprechend einem verbreiteten Usus wird REAL★8 erkannt und wie DOUBLE PRECISION behandelt.

Arbeitsweise

Mit ProFortran arbeitet man wie mit einem Fortran-System auf Großrechnern: Das Quellprogramm liegt in einer oder mehreren Textdateien vor (für sie ist unter ProFortran der Typ „.FOR" vorgesehen), diese werden mit einem Übersetzer in verschiebbare Maschinenprogramme umgewandelt (hier in Dateien vom Typ „.O") und dann zu einem lauffähigen Programm zusammengebunden (hier in eine Datei des Typs „.PRG"), wobei im letzten Schritt Bibliotheken auf benötigte Unterprogramme durchsucht werden können. Arbeiten kann man wahlweise unter GEM oder unter dem Befehlszeilen-Interpreter COMMAND.TOS.

Übersetzen

Der Übersetzer arbeitet zweistufig und wird durch den Aufruf des Steuerprogramms F77.PRG gestartet. Die erste Stufe übersetzt das Quellprogramm in einen kompakten Zwischencode, der in der zweiten Stufe in ein Maschinenprogramm verwandelt wird. Der Übersetzer arbeitet sehr schnell, wenn man doch einmal etwas länger warten muß, liegt dies an den langsamen Diskettenoperationen. Die mit einer RAM-Disk erzielte Geschwindigkeit ist jedoch beeindruckend.

Binden

Die übersetzten Programmteile müssen noch zu einem lauffähigen Programm zusammengebunden werden, wobei in der Regel noch Bibliotheken mit Maschinensprache-Programmen durchsucht werden müssen. Dies erledigt man mit dem Binder LINK.PRG. Die Namen der benötigten Programme und Bibliotheken müssen zuvor in eine Datei des Typs .LNK geschrieben werden, hierfür werden einige Musterdateien mitgeliefert. Je nachdem, ob man reine TOS-Programme schreibt (etwa zum Rechnen) oder ob man GEM-Routinen (etwa zur grafischen Ausgabe) benötigt, kann man zwischen zwei verschiedenen Systembibliotheken wählen.

Residente Laufzeitbibliothek

Die am häufigsten benötigten Unterprogramme wurden nicht in die Systembibliotheken aufgenommen, sondern zu einer speicherresidenten Laufzeitbibliothek zusammengefaßt. Diese wird einmal, am besten gleich nach dem Einschalten, in den Speicher geladen und verbleibt dort bis zum Ausschalten. Das zugehörige Programm heißt PRL.PRG. Die Laufzeitbibliothek wird übrigens auch von allen Programmen des Fortran-Systems benötigt.

Ausführen

Die gebundenen Programme erhalten den Typ .PRG und können wie alle anderen Programme gestaltet werden. Man kann sie auch mit einem Parameter aufrufen (in COMMAND.TOS oder als TTP-Anwendung), dieser kann im laufenden Programm über das Unterprogramm GETCOM geholt wer-

den. Außerdem kann ein laufendes Programm über das Unterprogramm EXECPG ein Tochter-Programm starten. Insgesamt bieten sich hier viele interessante Möglichkeiten, man kann sich etwa einen Satz Unix-ähnlicher Tools unter COMMAND.TOS erstellen.

Anhalten oder nicht?

Am Ende der Ausführung halten alle Programme des Fortran-Systems sowie alle damit erzeugten Programme an und geben die Kontrolle erst nach einem beliebigen Tastendruck ans Betriebssystem zurück. Dies kann unter GEM besonders wichtig sein, da sonst der vom Programm erzeugte Ausdruck (im Falle des Übersetzters etwa Fehlermeldungen) gleich wieder vom Desktop überschrieben würde. Beim Arbeiten unter COMMAND.TOS oder wenn das Programm keinen Ausdruck erzeugt, kann diese Warterei jedoch sehr lästig sein. Hier gibt es Abhilfe: Die Programme lassens ich mit dem ebenfalls mitgelieferten RCONFIG.PRG so modifizieren, daß die Wartefunktion an- und abgeschaltet werden kann.

Modifizierung des Systems

Der Übersetzer bietet einige der üblichen Optionen: Ausgabe der Umwandlungsliste, der Liste aller Variablen mit Typ und Speicherplatz, Aufspüren von nicht erklärten Variablen und vieles mehr. Alle diese Dinge lassen sich ein- und ausschalten, etwa beim expliziten Aufruf von F77.PRG. Andererseits kann man mit Hilfe eines Programms FCONFIG.PRG auch sämtliche Voreinstellungen im Programm F77 selbst, und zwar dauerhaft, verändern.

Der Binder LINK.PRG produziert automatisch einen Speicherbelegungsplan (load map), der jedoch selten benötigt wird und nur Zeit und Speicherplatz verbraucht. Diese Liste läßt sich durch Angabe des Parameters „-NOLIST" unterdrücken.

Optimaler Betrieb

Durch Anmelden im Desktop kann der Übersetzer so konfiguriert werden, daß er beim Anklicken eines **Quellprogramms** automatisch aufgerufen wird, siehe Bild 1. Unter COMMAND.TOS kann man sich die Arbeit mit BAT-Dateien wesentlich erleichtern. Bild 2 zeigt eine solche Da-

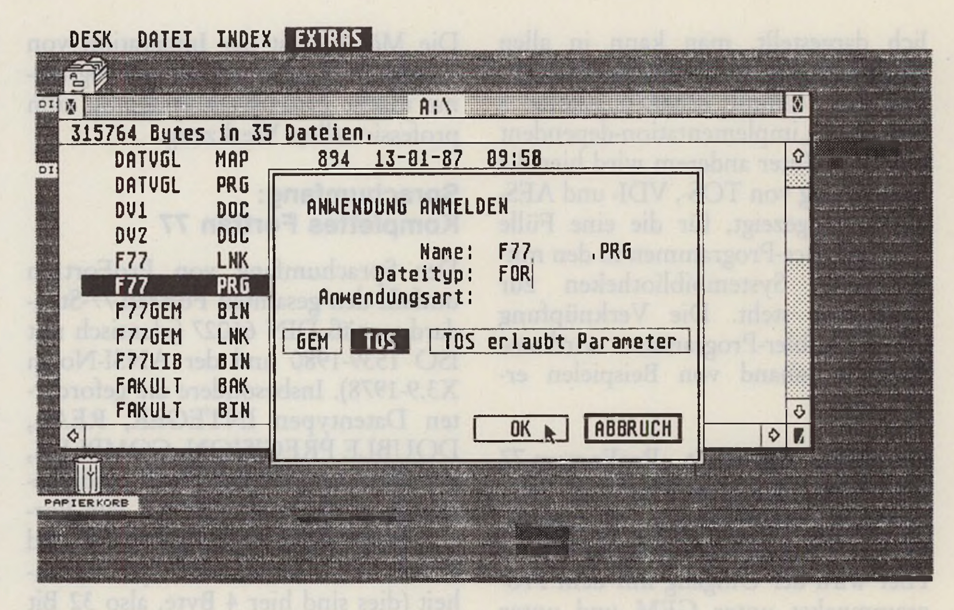


Bild 1: F77.PRG mit „Anwendung anmelden" vorbereiten

```
F77 %1
LINK %1 F77 -NOLIST
%1
```

Bild 2: Datei EXE.BAT zum Übersetzen, Binden und Ausführen eines Programms

```
Pro Fortran-77  -   Version mmg 1.1
Copyright (C) 1985 Prospero Software


Source filename - DATVGL

Default Options:

G - console output to LOG file ? (Y/N/.) N
I - range checks on subscripts ? (Y/N/.) Y
A - range checks on assignments ? (Y/N/.) Y
N - track source names & line numbers at run time ? (Y/N/.) Y
M - map ? (Y/N/.) Y
L - source listing ? (Y/N/.) Y
U - report undeclared variables ? (Y/N/.) Y
T - INTEGER means INTEGER*2 ? (Y/N/.) N
C - compact object code ? (Y/N/.) N
```

Bild 3: Eingabe aller Übersetzer-Optionen beim expliziten Aufruf

Bezugsquelle: Diverse Händler
oder Softline
Schwarzwaldstr. 8a
7602 Oberkirch

tei, die ein Programm automatisch übersetzt, bindet und anschließend ausführt. Der Programmname wird beim Aufruf als Parameter übergeben.

Programmpakete: NUMATH, PlotST, NAG

Inzwischen gilt es schon die ersten in ProFortran erstellten Produkte, so etwa das im Einsatz an Universitäten bewährte System NUMATH zur Verwaltung von Bibliotheken aus Fortran-Quellprogrammen oder das sehr leistungsfähige PlotST, eine Sammlung von Grafik-Unterprogrammen, wie sie auch auf Großrechnern zur Verfügung stehen. Prospero selbst hat schon eine Bibliothek mit den wichtigsten Programmen aus der bekannten und auf Großrechnern bestens eingeführten NAG-Bibliothek angekündigt.

Fehler und Unzulänglichkeiten

Der OPEN-Befehl zum Eröffnen einer Datei wird nicht umgehend ausgeführt, sondern erst beim ersten Schreib- oder Lesebefehl. Dadurch läuft eine eventuelle Fehlerabfrage, die mit den Parametern „ERR=“ oder „IOSTAT=“ im OPEN-Befehl arbeitet, ins Leere. Im Handbuch wird auf diesen Mißstand zwar hingewiesen (Motto: This is a feature, not a bug!), aber nicht etwa dort, wo der OPEN-Befehl erläutert wird, sondern bei den Implementierungshinweisen! Die hat man natürlich auch schon mal gelesen, aber wenn der Fehler dann viel später zum ersten Mal passiert...

Weiter gibt es beim Einlesen von Texten aus Dateien ein Problem: Ist die eingelesene Zeichenkette länger als 80 Zeichen, so erhält man nur irgendwelchen Unsinn. Bis zu 80 Zeichen funktioniert alles einwandfrei. Die getestete Version hat die genaue Bezeichnung mmg 1.12, möglicherweise ist dieser Fehler in einer neuen Version schon behoben.

Bewertung

Das Programmpaket ProFortran zielt auf einen Markt, der im englischen treffend mit „professional-at-home“ umschrieben wird. Hier werden sehr hohe Anforderungen an Kompatibilität mit gleichen oder ähnlichen Produkten auf Großrechnern und an das Preis-/Leistungsverhältnis gestellt. Durch die Einhaltung des vollständigen Fortran77-Standards und dem vergleichsweise niedrigen Preis wird ProFortran beiden Kriterien gerecht. Die Nachbildung der auch auf Großrechnern üblichen Arbeitsstufen (Übersetzen, Binden, Ausführen) und die Möglichkeit, auf verschiedenen Ebenen des Betriebssystems (TOS, GEM) zu arbeiten, machen den Umgang mit ProFortran für den Profi sehr angenehm, da er hier vertraute Arbeitsweisen und (Programmier-)Umgebungen vorfindet und auch umfangreiche Programmpakete bearbeiten kann.

Als wichtigste Erweiterung wäre ein Debugger zu wünschen, der die Fehlersuche bedeutend erleichtern würde. Sicher gibt es so etwas auch auf manchen Großrechnern nicht – aber warum sollten die leistungsfähigen Mikrocomputer nicht auch hier, wie schon der Textverarbeitung, ihren größeren Brüdern noch etwas vormachen?

Dr. Volker Kurz

Relax

SHANGHAI

Shanghai gehört zu jenen Spielen, die vor allem Konzentation und Logik erfordern. Mehrere Züge müssen im voraus durchdacht werden, wenn man die richtige Lösung finden will - und das ist nicht immer leicht.
Seine Grundzüge hat das Spiel von Mah-Jongg, einem über 3000 Jahre alten chinesischen Spiel, das Seeleute, Halunken und Könige gleichermaßen begeistert haben soll. Es besteht aus 144 Steinen, die pyramidenartig übereinandergeschichtet sind und einen Drachen darstellen sollen. Die Steine haben verschiedene Ornamente, die in sieben verschiedene Bedeutungsgruppen unterteilt sind. Der Spieler muß diejenigen Steine heraussuchen, deren Ornamente gleich sind und die nach links oder rechts bewegbar sind. Das sind immer die Steine, die am rechten oder linken Rand einer jeweiligen Pyramidenstufe liegen. Sie werden angeklickt und somit vom Spielfeld genommen. Da jeder Stein viermal vorhanden ist, beeinflußt die Auswahl oft das Spielende, das dann erreicht ist, wenn keine Pärchen mehr gebildet werden können. Wer gut kombiniert, der hat dann nur noch wenige oder sogar keine Steine mehr auf dem Brett.
Neben dieser SOLITAIRE-Variante gibt es verschiedene andere Arten. Beim PARTNERSCHAFTSSPIEL versuchen beliebig viele Spieler die Aufgabe gemeinsam zu lösen, im TURNIER kann mit Zeitlimit gespielt werden. Jeder Spieler erhält dabei dieselbe Ausgangssituation und muß in der vorgegebenen Zeit möglichst viele Paare 'wegklicken'. Die fünf besten Ergebnisse werden danach abgespeichert. Als letzte Spielart steht die HERAUSFORDERUNG zur Auswahl. Hier müssen zwei Spieler jeweils eine gewisse Zeit lang Punkt sammeln, bis kein Zug mehr möglich ist.
Die beiden letzten Varianten, in denen zwei Spieler gegeneinander spielen können, sind besonders spannend und motivierend, denn hier zeigt sich, wer gut kombinieren und nachdenken kann.

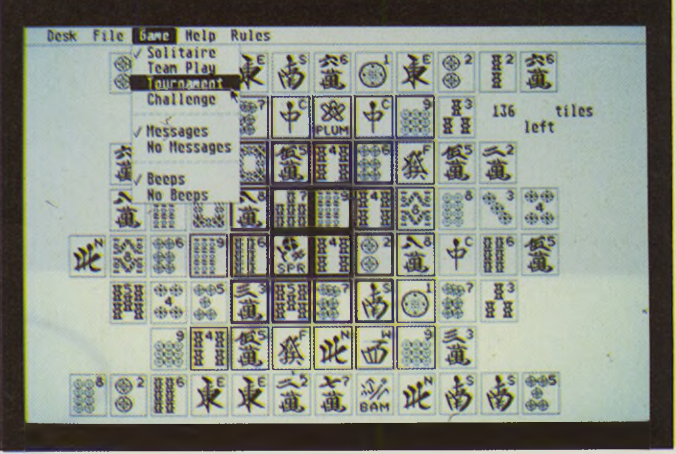
Bei den meisten Varianten stehen dem Spieler eine Reihe von Hilfsfunktionen zur Verfügung, die jedoch bei einem ehrlichen Spiel nicht eingesetzt werden sollten. Züge können zurückgenommen oder alle momentan möglichen angezeigt werden. Außerdem kann unter die Steine geschaut werden, um sich Klarheit über den Verbleib bestimmter Bilder zu verschaffen. Die Regeln der einzelnen Spiele und besondere Strategiehinweise können ebenfalls jederzeit aufgerufen werden.

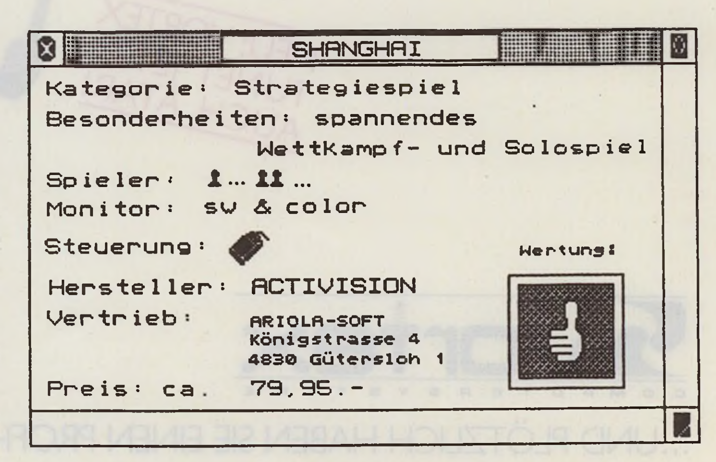


Die Grafik der einzelnen Spielsteine ist auf dem monochromen Bildschirm sehr gut zu erkennen, die zweidimensionale Darstellung des pyramidenartigen Aufbaus bereitet allerdings am Anfang einige Schwierigkeiten. Die höher liegenden Steine sind hier durch dickere Umrandungen gekennzeichnet. Auf dem Farbmonitor werden die Ebenen durch verschiedenfarbige Steine angedeutet. Nach dem Start des Spieles sitzt man vor den ungefähr 100 sichtbaren Steinen und sieht in dem Gewirr erst einmal überhaupt keine Paare. Mit der Zeit, wenn sich die einzelnen Symbole einprägen, werden immer mehr Kombinationen sichtbar und die Spannung steigt.

SHANGHAI ist ein faszinierendes Spiel, das jeden Spieler fest in seinen Bann zieht. Wer einmal damit angefangen hat, wird es so schnell nicht mehr weglegen und ständig nach Mitspielern für die TURNIER-Runden suchen.

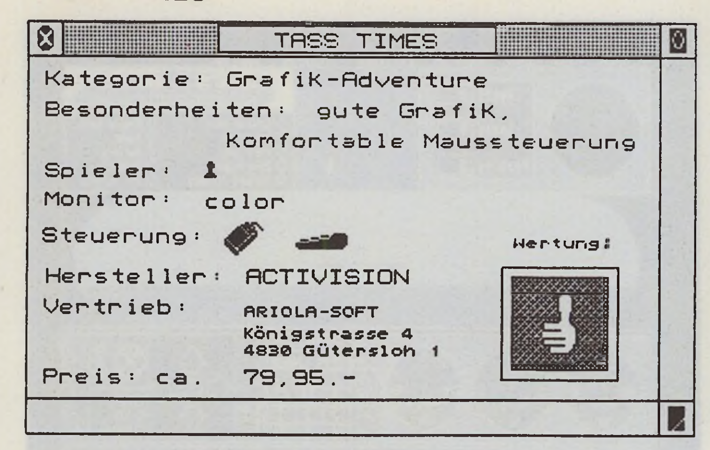





SHANGHAI

Kategorie: Strategiespiel
Besonderheiten: spannendes Wettkampf- und Solospiel
Spieler: 1... 2...
Monitor: sw & color
Steuerung:
Wertung:
Hersteller: ACTIVISION
Vertrieb: ARIOLA-SOFT, Königstrasse 4, 4830 Gütersloh 1
Preis: ca. 79,95.-

TASS TIMES



TASS TIMES

Kategorie: Grafik-Adventure

Besonderheiten: gute Grafik, Komfortable Maussteuerung

Spieler: 1

Monitor: color

Steuerung:

Wertung:

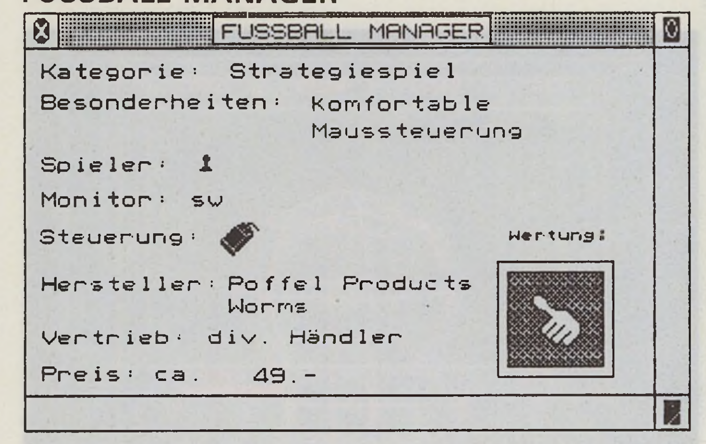
Hersteller: ACTIVISION

Vertrieb: ARIOLA-SOFT, Königstrasse 4, 4830 Gütersloh 1

Preis: ca. 79,95.-

TONETOWN ist der Alptraum jedes Fremden, der sich dieser Kultur nicht anpaßt. Grell und schrill ist hier alles: Die Bewohner, die Mode (nur 'Toppo Wear' zählt), die Frisuren ('Die-Cut' von Chaz) und die Musik, die aus allen Ecken dröhnt - der größte Hit ist 'TASS' von den Daglets.
Der Ablauf des Adventures erinnert stark an BORROWED TIME, denn auch hier hat man ständig einen Verfolger im Rükken (in diesem Fall "Franklin Snarl" von der Gattung der Krokodile), dessen Auftauchen gleichzeitig das Spielende bedeutet. Sinn des Spiels ist es, den verschwundenen Wissenschaftler Gamps zu finden, wobei nur Ennio, der legendäre Reporter-Hund, ein zuverlässiger Helfer ist. Wichtig ist in TONETOWN vor allem, nicht aufzufallen. Deshalb ist es ratsam, erst einmal neue Kleidung zu kaufen und danach sofort den Friseur aufzusuchen. Nun müssen nur noch die Indizien gesammelt werden, aber mit Vorsicht, denn der Verfolger ist Ihnen dicht auf der Ferse.
Die gute, teilweise bewegte Grafik von TASS TIMES ist, entsprechend der Handlung, schrill und skurril. Piktogramme (take, look, say usw.) können als Kommandos verwendet werden, indem man sie einfach mit der Maus anwählt. Auf die gleiche Weise ist es möglich, Objekte aus dem Szenenbild mitzunehmen, wenn dies vorgesehen ist. Der Spielstand kann jederzeit abgespeichert werden. Die Handlung von TASS TIMES ist originell, motivierend und nicht zu kompliziert, so daß man auch mit mittelprächtigen Englischkenntnissen gut vorankommt.

FUSSBALL MANAGER



FUSSBALL MANAGER

Kategorie: Strategiespiel

Besonderheiten: Komfortable Maussteuerung

Spieler: 1

Monitor: sw

Steuerung:

Wertung:

Hersteller: Poffel Products Worms

Vertrieb: div. Händler

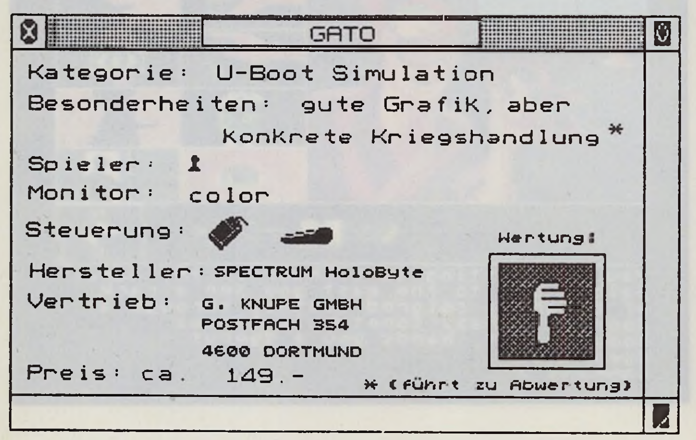
Preis: ca. 49.-

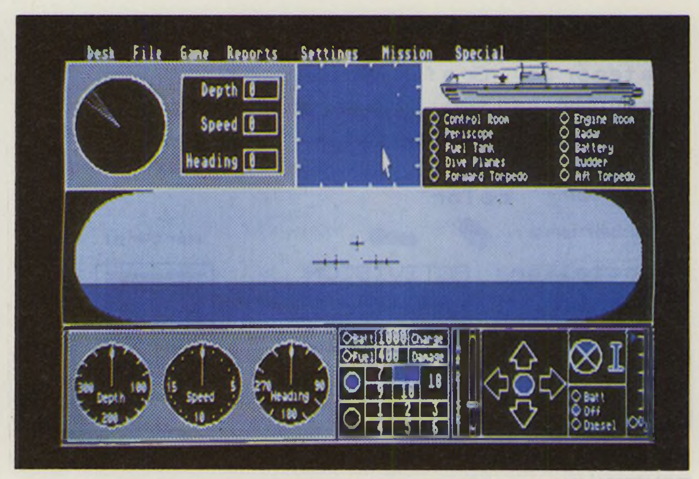
Das Managen eines Fußballvereins ist keine leichte Aufgabe: Die unterschiedlichsten Faktoren beeinflussen den Erfolg einer Mannschaft. Da wären zum einen die spielerischen Leistungen der einzelnen Spieler, die sich im Marktwert (von 250 000,- bis 2 Mio.) ausdrücken und den größten Einfluß auf den Ausgang einer sportlichen Begegnung haben. Weitere Punkte sind Öffentlichkeitsarbeit, Organisation, ärztliche Betreuung der Spieler, Trainerentlassung und die Förderung der Nachwuchsarbeit.

Spieler zu kaufen und verkaufen, bis die Mannschaft die optimale Spielstärke hat, ist eine der sichersten Strategien für den Weg an die Tabellenspitze. Doch für gute Spieler muß bekanntlich ein hoher Preis gezahlt werden. Nur, wer genügend Kapital hat, kann bei den Preisverhandlungen mithalten. Der einzige Weg, das Kapital zu vergrößern, sind hohe Einnahmen bei dem Heimspielen. Die Zuschauer strömen jedoch nur dann in die Stadien, wenn die Mannschaft auf einem hohen Tabellenplatz ist. Somit schließt sich der Kreis, der leicht zu einem 'Teufelskreis' werden kann, aus dem nur die besten Manager wieder herausfinden.

Die Zusammenstellung der acht Vereine, die an der Meisterschaft und der Pokalrunde teilnehmen, wird vom Spieler vorgenommen. Bis zu 10 laufende Spiele können jederzeit abgespeichert werden. Die Handlungen des Managers werden in der GEM-üblichen Weise vorgenommen, fast alle Aktionen können bequem und einfach mit der Maus kontrolliert werden. Leider kann der FUSSBALL MANAGER nur von jeweils einer Person gespielt werden - was niemanden daran hindert, sich das Management mit Freunden zu teilen. Auf jeden Fall ist das Spiel für Fußball-Fans zu empfehlen. Besonders dann, wenn sie mit dem Mißmanagement der Bundesliga-Vereine einmal aufräumen wollen.

GATO
- U-Boot Simulator

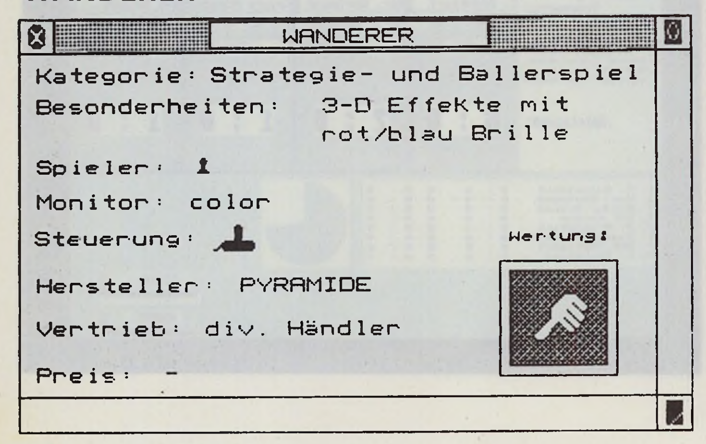




Als Kapitän eines U-Boots hat man es nicht leicht. Unzählige Instrumente müssen kontrolliert werden. Am Horizont tauchen bald feindliche Schiffe auf. Ihre Aufgabe ist es, diese Tanker oder Containerschiffe zu versenken, deren Bewegungen auf einer Übersichtskarte jederzeit verfolgt werden können. Mit Volldampf wird das Einsatzgebiet angesteuert, dann auf langsame Fahrt umgeschaltet und untergetaucht. Wenn auch noch das Sehrohr eingefahren ist, können die Schiffe nur noch mit dem Radar beobachtet werden. Nun beginnt der Bildschirmkrieg: Die Torpedorohre werden geladen und, sobald ein Schiff in Reichweite ist, abgefeuert. Wenn das Schiff nicht rechtzeitig abdreht, erscheint ein Feuerwerk am Rumpf und es verschwindet von der Bildfläche. Hektisch und gefährlich wird die Situation, wenn ein gegnerisches Schiff versucht, das U-Boot zu rammen, wenn es über Wasser ist. Andernfalls werden Wasserbomben abgeworfen, die das U-Boot mehr oder weniger zerstören. Dann gelingt es nur noch mit Geschick, aus der Umklammerung der nahenden Feindschiffe zu entkommen.

GATO ist in Gestaltung und Handhabung gut gelungen. Die Instrumente sind leicht kontrollierbar angeordnet und gut zu bedienen. Die Aktionen beziehen sich jedoch konkret auf Ereignisse des II. Weltkriegs ('vernichten Sie den japanischen Konvoy'). Inhalt und Sinn dieser Kriegs-Simulation (die Bezeichnung 'Spiel' wurde bewußt unterlassen!) erscheinen äußerst fragwürdig: Zu sehr wird eine Euphorie für das Versenken und Zerstören von Schiffen (mit Besatzung!) gefördert.

WANDERER



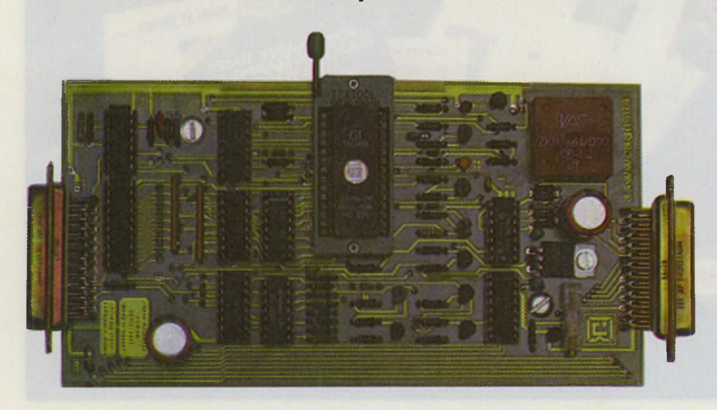
WANDERER ist das erste Spiel für den ATARI ST mit echten räumlichen Effekten (die dazu erforderlichen rot-blauen Brillen werden mitgeliefert!). Nach dem Starten befindet man sich im Demo-Mode des Spieles, der die dreidimensionalen Effekte vorstellt. Der Titelschriftzug saust dabei aus verschiedenen Richtungen über den Bildschirm, dreht Kreise und Loopings und verschwindet wieder. Danach kommen verschiedene Raumschiffe und vollführen ihre Kapriolen. Der 3D-Effekt ist allerdings recht gewöhnungsbedürftig. Erst nach einiger Zeit kann man die Objekte richtig erkennen. Doch bis zum Höhepunkt der Demo ist der Raumeindruck perfekt - die rotierende Netzgrafik einer Kugel kann bestaunt werden. Die hierbei erkennbaren Möglichkeiten der 3D-Darstellung sind wirklich faszinierend.

WANDERER nutzt diese Möglichkeiten leider nur ansatzweise, denn die nun folgenden Weltraumszenen, bei denen man von verschiedenartigen Raumschiffen angegriffen wird, sind etwas mager gestaltet. Der räumliche Eindruck kommt nicht sonderlich gut zur Geltung. Die Handlung des Spiels ist eine Mischung aus Kampfszenen und Pokerspiel; die eigentliche Aufgabe ist die Rettung einer entführten Katze. Drei verschieden schwere Wege führen zu diesem Ziel, wobei gekämpft, gespielt, nachgedacht und gehandelt werden muß.

Resümee: WANDERER erfüllt nicht die Erwartungen, die es als 3D-Spiel erweckt. Räumliche Effekte sind zwar vorhanden, die grafische Gestaltung ist jedoch recht einfach. Auch die Rahmenhandlung ist nicht geeignet, den Wert des Spiels zu heben.

Easyprommer, der Eprommer für Ihren ST.

- ★ Brennen aller gängigen EPROMtypen (2716 - 27513)
- ★ Brennen der modernen EEPROMs (X2804A - X28256A)
- ★ Auslesen der ROM-Typen 4732 - 47256 zum Verändern und Brennen
- ★ 5 verschiedene Brennalgorithmen (u. a. Auto-Modus)



- ★ Aufteilen der zu brennenden Software in High- und Lowbyte
- ★ Software voll GEM-unterstützt und Pull-Down-Menü gesteuert
- ★ Blitzschnelle Änderungen mit dem eingebauten Monitor
- ★ Vielfältige Austestmöglichkeiten
- ★ Eigene RAM-Disk, von der jegliche Software gebrannt werden kann
- ★ Schnelles Brennen über Parallel-Port
- ★ Eigener Druckertreiber, softwaremäßige Umschaltung zwischen Drucker und Prommer mit Betriebsanzeige
- ★ Update-Service für die Software bei Erscheinen neuer EPROMtypen
- ★ EPROMbank bis 512 KByte! in Vorbereitung
- ★ Ausführliches Handbuch

Händleranfragen erwünscht

Bezugsquelle: ST Computer Redaktion
Schwalbacher Str. 64
D-6236 Eschborn
Tel.: 06196/482158



Hiermit bestelle ich:

☐ Easyprommer (Fertiggerät) für DM 349,- + Versandkosten ☐ Vorauskasse

☐ Easyprommer (Bausatz) für DM 298,- + Versandkosten ☐ Nachnahme

Versandkosten: Inland DM 7,50, Ausland DM 10,—

Name: Vorname:

Straße: Ort:

Unterschrift:

ST Computer Redaktion

Infocom-Adventures

INFOCOM war eine der ersten Firmen die sich auf Textadventures spezialisierten. Die teilweise schon berühmten Adventures wie ZORK wurden vor einiger Zeit auch für den ATARI ST verfügbar gemacht. Inzwischen sind auch neue Adventures erschienen. Für uns ein Grund, einige dieser Spiele unter die Lupe zu nehmen.



Zur Verfügung standen uns die drei Produkte „LEATHER GODDESSES OF PHOBOS“, „A MIND FOREVER VOYAGING“ und „THE HITCHHIKER'S GUIDE TO THE GALAXY“. Gestaltung und Verpackung sind ansprechend. Klappt man die Packung auf, so findet man ein Heft, das außer der üblichen (ausführlichen) Programmbeschreibung bei 'LEATHER GODDESSES...' auch einen 3D-COMIC enthält! Man merkt, daß sich INFOCOM etwas einfallen läßt. Doch nun ins Detail:

LEATHER GODDESSES OF PHOBOS: Die Packung des als durchschnittlich schwierig eingestuften Adventures enthält einen Comic, der mit einer Rot-Grün-Brille dreidimensional erscheint. Außerdem wurde, als besondere Überraschung, eine Schnüffelkarte beigelegt: Rubbelt man an bestimmten Stellen der Karte, entsteht ein Geruch. Die Hintergrundgeschichte selbst ist nicht besonders anspruchsvoll und aus einem Comic entstanden: Die 'Ledergöttinen von Phobos' möchten die Menschheit unterjochen und die Erde zu ihrer Lustwelt machen. Sie sind dazu auserkoren, ihnen als „experimentelles Subjekt“ zu dienen. Ihre Aufgabe ist es, mit einem anderen 'Erdling', einer aus Alaska stammenden Person, aus den Fängen der Göttin zu entfliehen und die Dinge zu finden, die sie zur Vernichtung der Ledergöttinen benötigen.

Das Adventure ist sprachlich nicht immer jugendfrei. Die Warnung im Vorwort ist deutlich: „Bestimmte Teile der Story sind nicht für den Zugang von Kindern gedacht. Besonders die Teile, deren Inhalt Sex ist, sollte niemandem unter 18 Jahren zugänglich sein. Allerdings kann man drei verschiedene Spielstufen wählen: Zahm, Normal und Zügellos. Die Sätze werden dann, entsprechend Ihrer Moraleinstellung, 'entschärft'. Die Texte geraten aber nie obszön oder primitiv. Das Spielen des Adventures macht trotz einiger Nüsse, die es zu knacken gilt, viel Spaß, da es mit der Wahl der Sätze sehr gekonnt umgeht. Wer Spaß an einem interaktiven Comic hat, in dem er mal selbst den Helden spielen darf, dem kann zu diesem Adventure geraten werden. Für Anfänger in der Adventurewelt ist es allerdings zu schwierig.

HITCHHIKER'S GUIDE TO THE GALAXY: Hinter diesem englischen Titel verbirgt sich die Umsetzung der Science-Fiction-Serie PER ANHALTER DURCH DIE GALAXIS. Diese sehr humorvolle Geschichte, die wohl jedes Science-Fiction-Klischee auf die Schippe nimmt, ist besonders durch den gleichnamigen Mehrteiler im Fernsehen bekannt geworden. Das Buch, das bis dahin nur in den USA Erfolge feierte, wurde auch bei uns zum Bestseller. An dem Adventure hat der Autor des Buches, Douglas Adams, mitgearbeitet. Herausgekommen ist eine Menge an einfallsreichem Spielwitz. Bisweilen nimmt das Programm den Spieler sogar auf den Arm. Wem das Buch gefallen hat, wird auch Spaß an dem Spiel finden. Dennoch ist es keine schlichte Umsetzung des Buches, sondern mit einigen neuen Ideen ausgestattet. Als Arthur Dent bestehen Sie in Begleitung von Ford Prefect viele Abenteuer auf der Reise durch den Weltenraum. Das Adventure besticht durch seinen außerordentlichen Spielwitz und seine Ungewöhnlichkeit. Ein Tip noch für diejenigen Leser, die das Buch kennen: Glauben Sie nicht alles, was Ihnen Ihr Computer sagt, vergessen Sie Ihr Handtuch nicht und: DON'T PANIC!

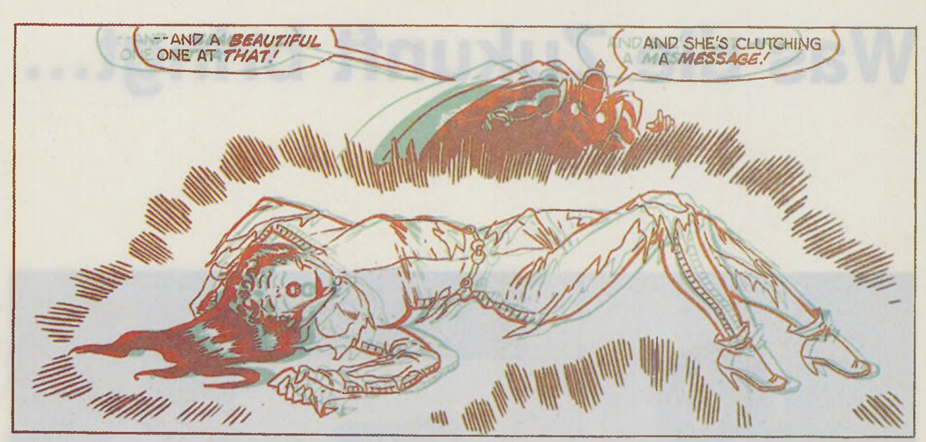
A MIND FOREVER VOYAGING: Man zählt das Jahr 2031. Die Welt ist ein Chaos. Die Arbeitslosenrate steigt, die Verbrechen nehmen immer mehr zu. Die Regierung hat einen Plan entwickelt, der den ökonomischen Frieden und die strengen moralischen

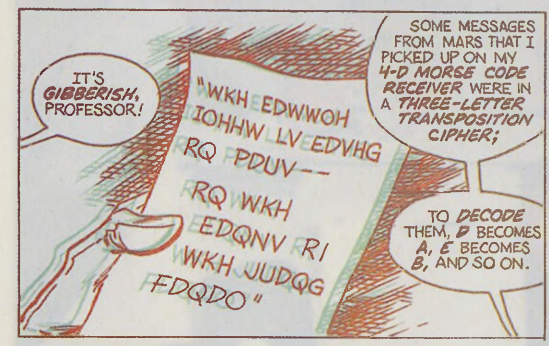
Werte der 50er Jahre vereinigt. Ob dieser Weg Frieden und Wohlstand sichern wird? Als erster Computer der Welt mit Bewußtsein und Intelligenz können nur Sie Plätze aufsuchen, die vorher noch nie gesehen worden sind... Schon der Hintergrund läßt vermuten, daß es sich um ein Adventure mit tiefgründigem Inhalt handelt. Es sollte nur noch gründlicher Erfahrung mit anderen Adventures angegangen werden. Lädt man das Programm, wird man mit Information regelrecht 'erschlagen'. Es gibt mehrere Modi zwischen denen man hin- und herschalten kann oder muß. Für manche Modi muß sogar ein bestimmter Code eingegeben werden, den man nur mit der mitgelieferten Code-Einstellscheibe herausfinden kann.

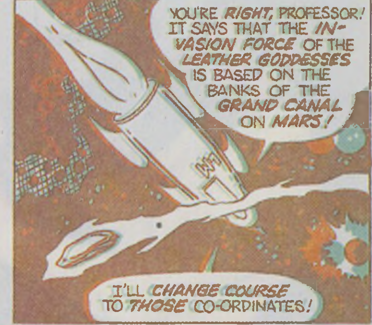
INFOCOM teilt seine Adventures in verschiedene Schwierigkeitsstufen ein: Anfänger, Standard, Fortgeschrittener, Experte. Die beiden Adventures 'LEATHER GODDESSES OF PHOBOS' und HITCHHIKER'S GUIDE TO THE GALAXY' sind als Standardadventures bezeichnet. Dennoch sind darin einige harte Nüsse zu knacken. 'A MIND FOREVER VOYAGING' fällt in die nächste Kategorie: Fortgeschritten! Wenn Sie einmal richtig herausgefordert werden möchten, kann dieses Adventure vorgeschlagen werden. Hier ist übrigens das häufig notwendige Nachladen störend (bei den anderen beiden Spielen nicht) was auf einem Rechner mit einem Speicher von mindestens 512 Kilobyte nicht vorkommen dürfte.

Resümee: Obwohl Grafikadventures auf den Markt vordringen, haben die reinen Textadventures ihre Berechtigung. Mancher Spieler bevorzugt sogar diese Art, da Bilder die Vorstellungskraft und Phantasie rauben. Gerade deswegen sind diese Spiele empfehlenswert: Sie regen noch die Phantasie an und nicht nur den Daumen. Am meisten machen sie mit ein paar Freunden Spaß, denn es kann schon so manchen Nerv kosten, bis man aus bestimmten Situationen herausgekommen ist. Gute Englischkenntnisse und ein dickes Wörterbuch sind vorausgesetzt.

Stefan Höhn





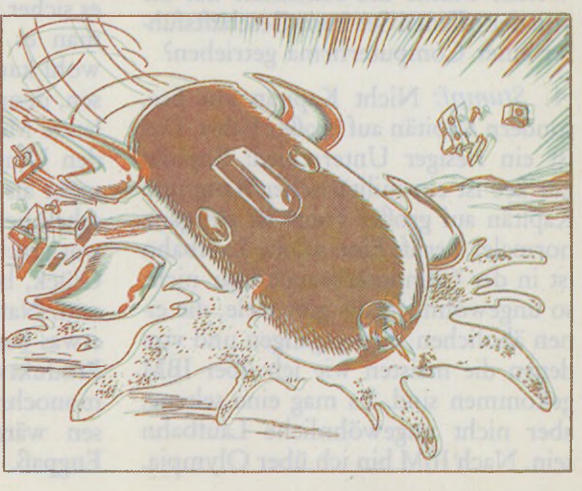




5







6

Was die Zukunft bringt...



Seit rund 20 Monaten ist der ATARI ST erfolgreich in Deutschland auf dem Markt. Wir haben uns mit Geschäftsführer Alwin Stumpf und Vertriebsleiter Klaus Peter Kuschke über das weitere Geschehen bei ATARI unterhalten. Wir konnten viele interessante Neuigkeiten erfahren, die wir Ihnen im folgenden Interview wiedergeben.

ST Computer: Herr Stumpf, seit über zwei Jahren sind Sie Geschäftsführer bei ATARI Deutschland. Wir haben gehört, Sie waren Kapitän zur See. Welche Macht des Schicksals hat Sie von der offenen See zum Geschäftsführer einer Computerfirma getrieben?

A. Stumpf: Nicht Kapitän zur See, sondern Kapitän auf großer Fahrt. Das ist ein riesiger Unterschied: Kapitän zur See ist ein militärischer Rang und Kapitän auf großer Fahrt ist ein ganz normaler Beruf. Eine solche Laufbahn ist in der Computerbranche gar nicht so ungewöhnlich. Es gibt viele, die einen ähnlichen Weg gegangen und von denen die meisten wie ich über IBM gekommen sind. Es mag eine seltene, aber nicht ungewöhnliche Laufbahn sein. Nach IBM bin ich über Olympia, wo ich den Schritt ins Management begonnen habe, und Commodore zu ATARI gekommen.

ST Computer: Die ATARI ST-Modelle sind zur CeBIT '87 eindreiviertel Jahre auf dem deutschen Markt. Ohne Zweifel sind die ST-Rechner im Moment die erfolgreichsten und leistungsstärksten PCs bzw. Home-Computer in Deutschland. Können Sie eine kurze Bilanz ziehen, da Sie schließlich von Anfang an dabei waren? Sind Sie zufrieden, oder hätte es hier und da besser laufen können bzw. sollen?

A. Stumpf: Nun, zurückschauend ist es sicher hervorragend gelaufen. Wenn man es von außen betrachtet, wäre wohl kaum etwas zu verbessern gewesen, denn ich weiß nicht, wann es das letzte Mal gewesen ist, daß eine Firma den Umstz in dieser Größenordnung mehr als verdoppelt hat. Es gibt wahrscheinlich nur zwei Firmen, die so etwas geschafft haben, und ich hatte das Glück, bei beiden Geschäftsführer zu sein. Natürlich hätte es im Herbst '86 etwas besser laufen können, wenn die Produktversorgung vor allem beim monochromen Monitor besser gewesen wäre, dies war der eigentliche Engpaß.

ST Computer: Es waren also nur Lieferschwierigkeiten und keine technischen Mängel?

A. Stumpf: Wir können hier nicht von technischen Fehlern reden, sondern eher von planerischen Unzulänglichkeiten. Wir hatten den Anteil von Farbmonitoren im Verkauf höher eingeschätzt. In Amerika verkaufen wir 90 Prozent Farbmonitore und hier ist es genau umgekehrt. In Deutschland verkaufen wir fast 95 Prozent Schwarzweiß-Monitore. Dies ist auch nicht verwunderlich, da die Stärken des ST unter anderem in der hohen Auflösung liegen. Mittlerweile können wir den monochromen Monitor uneingeschränkt liefern, allerdings gibt es nach der jüngsten Preissenkung auf DM 1998,– einen Engpaß beim 1040 ST/F.

ST Computer: Der ST ist von seiner Struktur her schwer in eine bestimmte „Klasse" einzuordnen. Seine Leistung könnte man mit einem Mikro-Rechner vergleichen, sein Design aber eher mit einem Home-Computer. Inwieweit hat diese Struktur die Einführung in den professionellen Markt beeinträchtigt oder sogar behindert?

A. Stumpf: In keiner Weise! Man muß feststellen, daß dies mit dem Gerät weniger zu tun hat als mit der Software. Den professionellen Markt muß man in einen mehr technisch-und einen mehr kommerziell orientierten Bereich unterscheiden. Zwangsläufig startet man im technischen Bereich, weil dort die Softwarefrage eine zweite Rolle spielt. Zum anderen liegt es in der Natur der Sache, daß Techniker eher die Leistung eines Computers beurteilen können als Kaufleute. Kaufleute kaufen mehr nach Verfügbarkeit, Namen, Image und nach der Software, während Wissenschaftler, gerade im technischen Bereich, eher bereit sind, solche Aspekte außer acht zu lassen und nur nach Leistung zu kaufen. Vor allem dieses Marktsegment sind wir von Anfang an stark angegangen. Dies war eine erfolgreiche Strategie. Mittlerweile ziehen viele, auch kommerzielle Bereiche nach. Die bestehende Software erlaubt zur Zeit den erfolgreichen Einsatz der ST-Rechner bis hin zu mittleren Betrieben.

Auf der anderen Seite ist mir die Klassifizierung der ST Rechner völlig gleichgültig. Eine solche Einstufung hat vielleicht eher etwas mit Image zu tun als mit der tatsächlichen Leistung eines

Rechners. Man kann heute die Rechner zwischen 2000 und 10 000 Mark für ein komplette Konfiguration in einen Topf werfen. Leistungsmäßig sind eher die niedrigpreisigen höher einzustufen als die der höheren Preisklasse. Die in der Mitte angesiedelten, also die gesamte PC-Welt, sind, wenn man es rein von der Computerleistung her sieht, viel weniger professionell als der ST. Auch wenn wir jetzt einen PC bringen, dann ist das technisch gesehen ein eklatanter Rückschritt. Auf der anderen Seite gibt es einen Markt dafür und es gibt Software dafür und wir sind mittlerweile so groß und nicht zuletzt durch die Aktienemission auch so liquide, daß wir auch in anderen Bereichen wachsen wollen und müssen.

ST Computer: Außer seiner technischen Leistung glänzt der ST durch sein Betriebssystem GEM. Wird die durch das GEM hervorragende Bedienungsfreundlichkeit ein Standard für die Zukunft werden? Andererseits läßt dieses Betriebssystem den Rechner ungewöhnlich häufig abstürzen. Wird ATARI die bekannten Fehler des Betriebssystems wie z. B. Unterstrich bei Dialogboxen oder auch XON/XOFF bei der RS-232 Schnittstelle verbessern?

A. Stumpf: Selbstverständlich wird ATARI die neuen Maßstäbe, die GEM gesetzt hat, weiterverfolgen. Sogar beim ATARI PC wird es eine GEM-Benutzeroberfläche geben.

Was Fehler im Betriebssystem anbelangt, so werden diese sicher behoben. Wie bekannt ist, verkaufen wir das TOS in sechs ROMs zu DM 148,– und ich könnte mir vorstellen, daß, sobald eine verbesserte Version vorliegt, diese zu einem ähnlichen Preis als Ersatzteil zu haben ist. Momentan ist diese Frage aber gegenstandslos, da es keine neue Version gibt.

ST Computer: Die Hardware des ST ist ein geschlossenes Konzept, daß sich nicht ohne großen Aufwand erweitern läßt. Dies geht sogar soweit, daß sich die Leistungsmerkmale des 68000-Prozessors nicht voll nutzen lassen. Da wäre z. B. die begrenzte RAM-Speicherkapazität bis maximal vier MByte. Wird ATARI in Zukunft ein offeneres Konzept bevorzugen, bei dem Erweiterungen, wie Koprozessoren etc. leichter realisierbar sind?

A. Stumpf: Von der Speicherkapazität her gesehen, halte ich maximal vier MByte für sinnvoll. Ein noch größerer Speicher, etwa 16 MByte, gehört in eine ganz andere Klasse, nicht mehr in die ST-Welt. Die Geschlossenheit des Systems, von der Technik her, ist ein Handicap. Es ist ein Zwang für uns, daß wir in Zukunft den Bus herausführen oder zumindest die Möglichkeit vorsehen. Das ist eine logische Weiterentwicklung.

ST Computer: Ist nun bei den neuen ST-Modellen ein Bus herausgeführt?

A. Stumpf: Da möchte ich mit konkreten Angaben ein wenig vorsichtig sein. Zur CeBIT werden wir genaueres sagen.

ST Computer: Werden die neuen Mega-Modelle softwarekompatibel zu den bisherigen Rechnern sein?



A. Stumpf: Soweit mir bekannt ist, gibt es keine Kompatibilitätsschwierigkeiten, es sei denn, der RAM-Speicher reicht nicht aus.

ST Computer: Erfahrungsgemäß altern gerade in der Computerbranche Rechnermodelle sehr schnell und werden von neueren und besseren Maschinen abgelöst...

A. Stumpf: Der ST wird mit Sicherheit aufgrund seiner modernen Architektur sehr langlebig sein. Vergleichbar mit dem Urmodell von IBM, dem 360, den hat es ja auch in verschiedenen Varianten über Jahrzehnte gegeben.

ST Computer: Nicht nur immer größere Speicherkapazitäten, sondern auch höhere Bildschirmauflösungen, eine große Anzahl von Anschlüssen für moderne Peripheriegeräte usw. spielen bei der Kaufentscheidung und Anwendung eine große Rolle. Zweifellos ist der verwendete Prozessor ein entscheidender Faktor für den massiven Einsatz eines Computers. Wie sieht ATARI die Möglichkeit, neue Modelle auf der Basis eines 32-Bit-Prozessors preisgünstig auf den Markt zu bringen?

A. Stumpf: Bei einem Rechner auf Basis eines 32-Bit-Prozessors wird es sich um einen ganz neuen Computer handeln, der dann mehrplatzfähig ist oder als Workstation einzusetzen ist. Ein solcher Rechner ist dann nur noch bedingt ST-kompatibel, da es sich dann um eine ganz neue Leistungsklasse handelt.

ST Computer: Es handelt sich also bei den neuen ST-Mega-Modellen um ST-Rechner in einem neuen Gehäuse mit einer im Druckpunkt verbesserten Tastatur und vergrößerter Speicherkapazität beim Mega ST2 und Mega ST4?

A. Stumpf: Ja, da sind ein paar Zugeständnisse, um besser in den kommerziellen Bereich einzudringen. Man muß da ein bißchen die Spielregeln einhalten, und das hatten wir mit dem 1040 ST/F nicht so ganz getan.

ST Computer: Gibt es weitere Unterschiede der neuen Modelle z. B. beim 1040 ST/F, wie werden die Preise sein und wann sind sie verfügbar?

A. Stumpf: Die neuen Rechner haben ein komplett neues Platinen-Layout, so daß sich z. B. der Blitter-Chip leicht nachrüsten läßt, falls er nicht schon eingebaut ist. Ferner lassen sich diese Modelle (Mega ST1 und Mega ST2) leicht auf vier MByte RAM-Speicher hochrüsten. Ob alle drei Modelle auf den deutschen Markt kommen oder ein Modell wegfällt, ist noch nicht entschieden. Die Preise stehen fest, ich möchte sie aber noch nicht veröffentlichen. Zum Liefertermin läßt sich sagen, daß wir mit der Produktion nach der CeBIT beginnen werden, d. h. es wird Ende Mai, bis sie erhältlich sind. Auf jeden Fall werden wir den Mega ST vor dem PC bringen, auch um zu zeigen, welches für uns das wichtigere Gerät ist.

ST Computer: Mittlerweile gibt es weltweit ein riesiges Softwareangebot für die ST-Rechner. Die Qualität der Software erreicht jedoch häufig nicht das gewünschte Maß an Professionalität, die der Anwender sich für einen solchen Rechner wünscht. Die Programmiersprchen bilden hier eine glückliche Ausnahme. Große Softwarehäuser, wie z. B. Ashton Tate, Microsoft oder

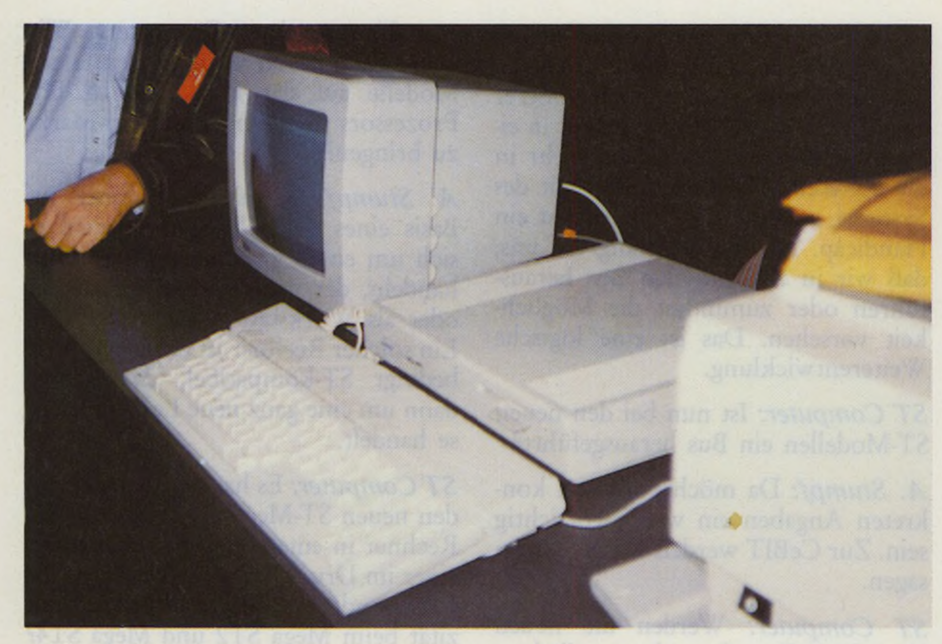


Bild 3: Mega Modell

Borland schreiben nur sehr zögernd Software für den ST. Meinen Sie, daß sich dieser Mißstand in Zukunft ändern wird?

A. Stumpf: Daß es qualitative und professionelle Software für den ST gibt, ist heutzutage unbestritten. Deutliche Beispiele sind das relationale Datenbanksystem Adimens oder die nun fertiggestellte Textverarbeitung 1st Word Plus. Selbstverständlich gab es am Anfang einige Produkte, vor allem im kommerziellen Bereich, die nicht den gewünschten Anforderungen der Anwender entsprachen. Dies ist aber bei einem völlig neuen Rechner, der eine sehr erfolgreiche Entwicklung von Beginn an hatte, eine normale Entwicklung. Schließlich bemühen sich die Softwareentwickler, das Vakuum an geforderten Produkten schnell zu füllen.

Was die großen Softwarehäuser anbelangt, so hängen die Interessen dieser Firmen in direktem Zusammenhang mit der Entwicklung eines Rechners im Markt. Dazu muß man sagen, daß der Verkauf der ATARI ST-Serie in Europa verhältnismäßig größer ist als in den USA. Ein weiterer wesentlicher Faktor ist die große Verbreitung der IBM-Rechner bzw. kompatibler Systeme, wodurch das Interesse an nicht kompatiblen Maschinen in den USA geringer ist.

Mittlerweile hat sich diese Meinung durch den massiven Verkauf der ST-Geräte geändert, so daß es das bekannte Textverarbeitungssystem MS-Word der Firma Microsoft schon bald für den ST geben wird.

ST Computer: Aber in Deutschland sieht die Situation etwas anders aus. Hierzulande bieten die Softwarehäuser wie nie zuvor Produkte für den ST an. Wie erklären Sie sich, daß der ATARI ST in Deutschland eine so große Rolle spielt?

A. Stumpf: Wie schon gesagt, ist der Erfolg der ST-Serie in Europa und vor allem in Deutschland groß. Über 100 000 ST-Rechner sind in Deutschland verkauft, was zeigt, daß die Deutschen sehr sensibel auf moderne Technik reagieren und daß Kaufkraft vorhanden ist. Deutsche Softwarehäuser haben diesen Markt rechtzeitig erkannt und liefern qualitativ hochwertige Produkte.

Der ATARI PC

ST Computer: Mit Erstaunen haben wir gehört, daß ATARI einen MS-DOS-Rechner auf den Markt bringt. Ist das eine Abfuhr an den schon gezeigten MS-DOS Emulator für den ST?

K. P. Kuschke: Nein! Ganz klar nein, von Jack Tramiel wurde in Las Vegas bestätigt, daß der Anschluß der ST-Modelle zur MS-DOS Welt kommen wird und zwar in Form eines Hardware-Emulators.

A. Stumpf: Wir müssen zugeben, daß bei der Hardwareemulation – eine Softwareemulation liefert bereits 80-prozentige Kompatibilität – technische Schwierigkeiten aufgetreten sind, die am Anfang unterschätzt wurden. Zu diesen Schwierigkeiten gehört z. B. die Bildschirmdarstellung der MS-DOS-Programme auf dem SM124 mit seinen 71 Hz Bildwiederholfrequenz. Nicht ungelegen kommt uns jetzt der ATARI PC Clone, von dem sicher einige Bestandteile im Emulator Verwendung finden.

ST Computer: Jack Tramiel hat mit der ST-Serie bewußt keinen IBM kompatiblen Rechner gebaut und wollt



Bild 4: Der ATARI PC

emit seinem TOS-Betriebssystem einen Meilenstein gegen das verbreitete MS-DOS setzen. Ist damit die Einführung des ST im professionellen Bereich gescheiter?

K. P. Kuschke: Keineswegs! ATARI ist ein Unternehmen, das wächst, das sogar schnell wächst und in Zukunft noch schneller wachsen soll. ATARI ist seit November '86 an der amerikanischen Börse notiert und die Entwicklung der Aktien ist außerordentlich erfolgreich. Wenn man über Computervermarktung redet, darf man den Gesamtmarkt nicht außer acht lassen. Es existiert ein MS-DOS Markt und ein nicht MS-DOS-kompatibler Markt. Der nicht kompatible Markt hat einen Anteil von 30 bis maximal 35 Prozent der gesamten PC-Welt. Von diesem nicht kompatiblen Markt belegt der ATARI ST etwa 70 Prozent! Insofern ist die Expansionsmöglichkeit in diesem Marktbereich eingeschränkt, wenn ATARI nur in diesem Bereich bleibt. Da ATARI aber weiter und schneller wachsen möchte, muß man den MS-DOS-kompatiblen Markt angehen. So gesehen, ist es eine rein kaufmännische Entscheidung gewesen, den ATARI PC zu entwickeln.

ST Computer: Wie kompatibel ist der ATARI PC, was wird er kosten und was beinhaltet der Lieferumfang?

K. P. Kuschke: Der ATARI PC wird eine hervorragende Grundausstattung haben un dzu einem marktgerechten Preis erscheinen. Der ATARI PC ist mit 512 KByte RAM-Speicher, EGA-Karte, einem Mausport sowie Maus ausgestattet. Um eine möglichst hohe Kompatibilität zu erreichen, laufen Verhandlungen mit drei verschiedenen Softwarehäusern, die das Betriebssystem (BIOS) für den PC schreiben sollen.

ST Computer: Wird der ATARI PC unter dem Preis von Schneider PC liegen, also unter DM 2000,–?

K. P. Kuschke: Der Preis ist noch nicht endgültig ausdiskutiert. Die Preise von 499,– Dollar **ohne** Monitor und 699,– Dollar mit hochauflösendem Grün-Monitor wurden genannt. Wobei diese Preise amerikanische Preise sind, die keine Steuern beinhalten.

A. Stumpf: Der ATARI PC wird in Deutschland wahrscheinlich nur in der Version mit dem monochromen EGA-Monitor für unter DM 2000,– erscheinen.

ST Computer: Die CeBIT ist immer ein Schauplatz aller Neuigkeiten. Was wird ATARI außer dem PC neues zeigen?

A. Stumpf: Zur CeBIT wird mindestens eines der neuen Mega-Modelle zu sehen sein. Weiterhin wird der ATARI-Laserdrucker präsentiert. Es wird ein sehr preiswerter Laserdrucker, wobei die notwendige „Intelligenz" und der benötigte RAM-Speicher vom Rechner zur Verfügung gestellt wird. In Amerika wurde ein Preis zwischen 1400 und 1700 Dollar, je nach Ausstattung, genannt. In Deutschland wird der Drucker voraussichtlich Anfang Juni auf den Markt kommen.

ST Computer: Herr Stumpf, wir haben bis jetzt schon viel Interessantes erfahren. Abschließend wären da noch ein paar Produkte, von denen man teilweise bei der letzten CeBIT etwas gehört hatte. Dazu gehört z. B. eine 10-MByte-Floppy, die auch als Streamer für die Festplatte Verwendung finden könnte, oder auch ein ganz neuer Rechner mit wesentlich höherer Bildschirmauflösung und Unix-Betriebssystem. Oder wie steht es um eine ATARI-Produktionsstätte in West-Berlin bzw. Europa?

A. Stumpf: Eine 10-MByte-Floppy ist nach wie vor im Gespräch, allerdings gibt es auch interessante Alternativen zu einem solchen Massenspeicher. Zum Beispiel gibt es schon sehr handliche und robuste Fest-Wechselplatten, wobei die Preise, wie bei optischen Massenspeichern, noch unerschwinglich sind. Ein neues Modell auf der Basis eines echten 32-Bit-Prozessors aus der Motorola Familie ist in Planung. Als Betriebssystem wird dieser Rechner ein in GEM eingebundenes Unix haben, obwohl wir damit zur Zeit nicht hundertprozentig zufrieden sind. Ein Prototyp kann eventuell noch im Herbst dieses Jahres vorgestellt werden, aber wir haben es nicht besonders eilig, ein solches Modell einzuführen.

An eine Produktionsstätte in Europa ist derzeit, bei dem niedrigen Dollarkurs, nicht zu denken. Wir produzieren in Taiwan bzw. den USA wesentlich günstiger.

ST Computer: Vielen Dank für das Interview und weiterhin viel Erfolg.

Der ATARI PC

Besondere Merkmale:
Höhere Geschwindigkeit durch umschaltbare Taktfrequenz [8 MHz und 4,77 MHz (original IBM)].
Durch den Einsatz sogenannter Custom-Chips ist der ATARI PC EGA, CGA, Hercules und monochrom kompatibel.
Ein zusätzlicher Bildschirmspeicher von 256 KByte stellt den eingebauten 512 KByte RAM-Speicher vollständig für Programme zur Verfügung.
Eingebauter Mausport u. Maus inkl.
Erweiterungsmöglichkeit bereits vorgesehen, über zusätzliche geplante Erweiterungs-Box.
Sockel für Arithmetik Co-Prozessor 8087 vorhanden.
Eingebaute Standard-Schnittstellen (parallel und serielle RS 232C).
Hochauflösender EGA-Monitor.

Technische Daten:

CPU
Haupt-Prozessor: 8088
Taktfrequenz: 8 MHz /4,77 MHz
Vorbereitet für 8087

Speicher
Standard RAM: 512 KByte
Erweiterbar bis: 640 KByte
zusätzl. Bildspeicher: 256 KByte

Massenspeicher
Diskettenlaufwerk: 5 1/4" DS/SD
Kapazität: 360 KByte

Bildschirmdarstellung
EGA
CGA
Monochrom
Hercules
Monochrom Monitor
Spalten x Linien: 80 x 25
max. Auflösung: 720 x 348
Farb Monitor
Spalten x Linien: 80 x 25
max. Auflösung: 640 x 350
Anzahl Farben: 64
davon gleichzeitig darstellbar: 16

Anschlüsse
Parallel
RS 232C seriell
Maus-Port
Maus: in Lieferumfang enthalten

Tastatur
Gestaltung wie: PC/XT
Anzahl Tasten: 84

Preis: unter DM 2000,–
mit monochrom Monitor

(UB + MM)

Direktorenteam im Test:

Art- und Film-Director unter der Lupe



Art Director Einschaltbild



Film Director Einschaltbild

Es war ein regnerischer Herbsttag. Der Atari-Stand auf der Orgatechnik in Köln wurde nun schon das dritte Mal umrundet, als uns ein bisher dunkler Farbmonitor ins Auge fiel, der nun hell erleuchtet war. Vorsichtig näherten wir uns, bis uns bewußt wurde, das es sich hier um etwas ganz besonders handelte, das bisher nur als Gerücht und von Demos in ungarischer Sprache bekannt war. Daß unser Test erst heute erscheint, hat seinen Grund: Die zunächst angegebenen Lieferzeiten von etwa zwei Wochen wurden erheblich überschritten. Dennoch reichte die Zeit nicht aus, eine Bedienungsanleitung wenigstens in gutem Englisch zu schreiben.

Bei Film- und Art-Director handelt es sich um Grafikprogramme, die nur in der niedrigsten Auflösung (mit Farbmonitor) betrieben werden können. Der Film-Director benötigt zusätzlich einen ST mit 1 MB RAM und Tos im ROM.

Zunächst zum Art-Director: Dieses Programm verfügt über Features, wie sie kein anderes Grafikprogramm auf dem ST vorweisen kann. Nicht einmal „Deluxe Paint" auf dem Amiga bietet mehr (außer den zusätzlichen Farben). Nach dem Laden präsentiert sich das Programm auch erst einmal ungewohnt: Statt einer Menüleiste gibt es eine schmale Dialogbox auf dem Bildschirm. Sie enthält einige Malerwerkzeuge. Nichts besonderes also. Doch man kann diese Dialogbox verschieben. Dazu dient ein unscheinbares Feld in der linken unteren Ecke. Aber wo sind denn endlich die besonderen Funktionen? Der an DEGAS gewöhnte Zeichner probiert sofort die rechte Maustaste – siehe da, eine Menüleiste erscheint. Im Desk-Menü findet man gleich etwas spannendes. Dort leuchtet dem Anwender verheißungsvoll „Advanced Features" entgegen. Aber zügeln wir unsere Neugier erst mal ein wenig und schauen wir uns die „Toolbox", den Werkzeugkasten, wie die Dialogbox genannt wird, einmal näher an. Als erstes probieren wir das Feld links oben aus, das der „Close-box" eines Fensters so ähnelt – und schon ist die Toolbox weg. Mit F10 wird sie wieder sichtbar gemacht. Rechts von dieser Close-Box findet man das „Zoom-Window". Darin sieht man im Normalbetrieb die Umgebung des Malwerkzeuges in 2-facher Vergrößerung, im Lupenbetrieb den Lupenausschnitt in Originalgröße.

Die Vergrößerung der Lupe läßt sich zwischen 3- und 12-fach einstellen. Während der Darstellung einer Lupenseite und überhaupt bei jeder Aktivität des Programmes ertönt im Lautsprecher ein gefährliches Brummen. Gut, daß der Monitor einen Lautstärkeregler hat!

Die nächste Toolbox-Funktion ist die Schere. Damit kann man Bildschirmteile ausschneiden, die damit automatisch zum Pinsel werden. Man kann mit so gewonnenen Ausschnitten, die bis zu einem Bildschirm groß sein dürfen, beliebig malen. Für alle Zeichen- und Füllfunktionen kann so ein ausgeschnittener Pinsel, ein Stift oder einer der vordefinierten Pinsel verwendet werden. Außerdem steht noch eine Sprühdose mit veränderbarer Stärke und Größe zur Verfügung. Auch mit ausgeschnittenen Pinseln kann gesprüht werden. Leider ist die Sprühdose auch in ihrer stärksten Einstellung etwas schwach auf der Brust. Neben dem Sprühdosen-Symbol finden sich in der Toolbox je nach Funktion ein bis zwei Regler, die für eine Geschwindigkeits- und Stärkeneinstellung verwendet werden können (Sprühdose, Malgeschwindigkeit und ähnliches).

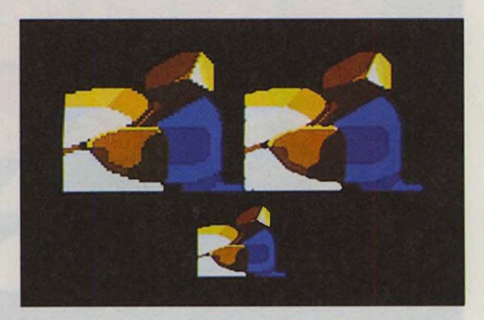
Bei der Füllfunktion wundert man sich zuerst – wo sind denn nur die Füllmuster? Doch beim Art-Director geht es anders. Zuerst einmal zum Verständnis: Der Art-Director besitzt zwei Malseiten, zwischen denen nach Belieben kopiert werden kann. Als (Füll-) Muster wird entweder der aktuelle Pinsel verwendet oder eine vorgegebene Farbe. Um „gewöhnliche" Füllmuster wie in anderen Programmen verwenden zu können, steht eine besondere Menüfunktion zur Verfügung. Mit ihr lassen sich 16★16 Punkte große Ausschnitte aus einem Bild, die in einem Raster auf dem Bildschirm liegen, auswählen. Der Witz dabei ist, daß diese „Pattern" wie gewöhnliche Bilder geladen werden. Man kann also spezielle Bilder, die die gewünschten Pattern enthalten, malen und abspeichern – oder auch eine mitgelieferte Palette von 104 Mustern verwenden. Zur Erzeugung von Füllmustern legt das Programm also ein imaginäres Raster von 16★16-Punkt-Blöcken über eine Bildschirmseite, aus dem der Benutzer dann eine Position auswählen kann. Ob der Bildschirm zu diesem Zeitpunkt gar nichts, ein beliebiges Bild oder vorgemalte Pattern enthält, ist der Funktion dabei gleichgültig. Natürlich ist es bei einem beliebigen Bild sinnvoller, einen Ausschnitt mit der Scheren-Funktion als Füllmuster auszuschneiden, weil man dann nicht an das Raster gebunden ist. Bei Verwendung von vorgefertigten Mustern ist diese Funktion aber sehr nützlich; man muß nicht jedes Pattern einzeln abspeichern, wie z. B. die Füllmuster von DEGAS.



Die Venus von Milot



Verzerrte Venus.



Das „Round Off"-Kommando:

Das gleiche Prinzip wird auch für die Schrifterzeugung benutzt. Mit der Textfunktion in der Toolbox kann nur die Atari-Standardschrift erzeugt werden. Man kann Text auch über mehrere Zeilen eingeben und dann den gesamten Textblock wie einen Pinsel verwenden. Will man andere Schrifttypen verwenden, muß man anders, ähnlich wie bei der Patternerstellung, vorgehen. Der Font wird nämlich ebenfalls wie ein gewöhnliches Bild erzeugt und gespeichert. Die Textfunktion wählt dann nur einzelne Blöcke nach einer Maske aus. Wie funktioniert das? Zuerst lädt man den gewünschten Font in die zweite Bildschirmseite – wie man sieht, ist gleichzeitiges Arbeiten auf zwei Bildschirmen praktisch unmöglich, denn man braucht die eine Seite meist als Zwischenspeicher. Dann wählt man die richtige Schriftgröße im Fontmenü an und beginnt zu schreiben. Den fertiggestellten Textblock übernimmt man dann als Pinsel und kann ihn somit beliebig auf dem Bildschirm plazieren oder auch damit malen.

Das Erstellen neuer Fonts ist im Prinzip ebenfalls einfach. Man lädt die (vorgefertigte) Maske für die entsprechende Schriftgröße auf den Bildschirm, „malt" seinen Font und speichert das neue Bild wieder ab, nachdem man die Maske gelöscht hat. Wirklich supereinfach – das Dumme ist nur, daß zumindest auf der Diskette unseres Testexemplares nur die Maske einer einer Schriftgröße enthalten ist, was die Erstellung anderer Schriftgrößen schwierig machen wird. Man kann zwar die Masken für die anderen Größen nach den vorhandenen Fonts nachzeichnen, aber das dürfte ziemlich viel Arbeit sein. Obwohl dieses System etwas gewöhnungsbedürftig ist, ist es eine gute Idee: Man spart sich den sonst notwendigen Fonteditor, dessen Funktionen sich mit der Lupe gut simulieren lassen. Die Fonts werden als normale Bilder gespeichert, was wirklich sehr bequem ist. Außerdem können Fonts mit allen Farb- und sonstigen Funktionen des Programms erstellt werden, was z. B. mehrfarbige Fonts erlaubt.

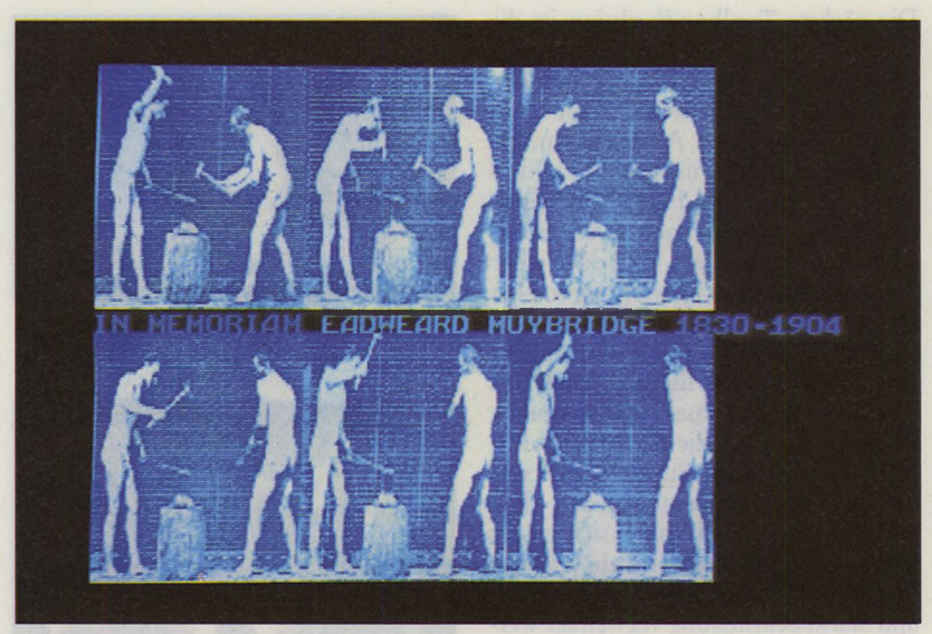
Weitere Funktionen der Toolbox betreffen das Zeichnen. Mit den Symbolen für „Smooth-" und „Rough-Lines" läßt sich das Zeichnen von Freihandzeichnungen erleichtern. In der „Rough-Lines"-Einstellung wird der Pinsel nicht an jeder Mausposition neu gezeichnet, sondern nur an jeder zweiten oder dritten. Besonders beim Zeichnen mit großen Pinseln bringt das Vorteile. Eine Raster-Funktion, die alle Zeichenfunktionen an ein frei definierbares Raster „klebt", existiert ebenso wie eine Fenster-Funktion, mit der es möglich ist, einige komplexe Funktionen (die später erläutert werden) auf einen Bildschirmausschnitt zu begrenzen.

Abgerundet werden die Toolbox-Funktionen durch die Farbenpalette, eine Möglichkeit, den Bildschirm zu löschen sowie den Farbpaletteneditor. Der Art-Direktor bietet die Möglichkeit, bis zu acht Farbpaletten zu definieren. Dafür gibt es auch Kopierfunktionen, die das Erstellen von Varianten einer Ausgangspalette sehr erleichtern. Mit Hilfe der achte Paletten kannman

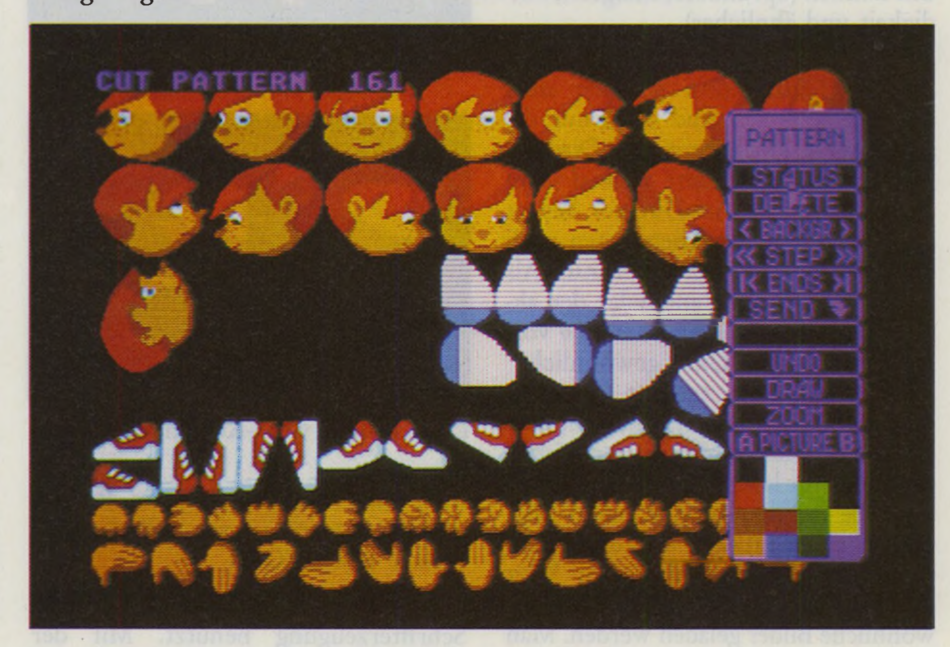
auch eine Farbenanimation, wie sie von Neochrome bekannt ist, verwirklichen. Hier werden allerdings nicht die einzelnen Farben der Palette durchgesteppt, sondern die ganzen Paletten werden der Reihe nach ausgetauscht. Dabei kann man für jede der Paletten eine Zeit angeben, in der sie verwendet werden soll. Schließlich kann jede Bildseite für sich beliebig über den Bildschirm „gerollt“ werden, wobei über den Rand geschobene Teile am gegenüberliegenden Rand wieder auftauchen. Man kann auch einfach nur vertikal zeilenweise den Bildschirm verschieben, wobei dann die andere Bildschirmseite zeilenweise ins Bild gelangt.

Werfen wir nun einen Blick auf die Menüleisten. Unsere Neugier gilt zuerst den „Advanced Features“: Da findet sich zuerst „Make Brush“, eine Reihe von Funktionen, die eine Erweiterung der Scheren-Funktion darstellen. Alle diese Manipulationen werden bei der Erstellung der Pinsel angewandt, können also nicht nachträglich verwendet werden. Ganz oben findet man „Rescale“ und „Stretch“ die es erlauben Bildausschnitte zu vergrößern, wobei Stretch rechteckige Verzerrungen erlaubt. Mit „Distort“ kann man Bildteile in beliebige Viereckformen verzerren. „Rotate“ dient zum Verdrehen von Ausschnitten. Die „Bend“-Funktionen erlauben das horizontale oder vertikale Verbiegen von Bildteilen, als würden die Ausschnitte auf eine Tonne geklebt, während „Bulge“ den Ausschnitt auf oder in eine Kugel projiziert. Schließlich kann man mit der „Perspect“-Funktion einen beliebigen Bildteil echt perspektivisch verzerren, wobei man Fluchtpunkt, Blickwinkel und Abstand wählen kann. Mit diesen Funktionen macht das Malen Spaß.

Vor allem die „Perspekt“-Funktion hat es mir angetan. Aber mit allen Funktionen kann man herrliche Dinge tun. Die Verzerrung der Gesichtszüge von digitalisierten Bildern ist nur ein Beispiel. Etwas gewöhnungsbedürftig ist das Verfahren: Man wählt erst den Bildteil aus, der ausgeschnitten werden soll, dann die Parameter der Verzerrung (z. B. für Stretch die Größe des Pinsels nach der Vergrößerung), dann erst wird der Bildteil ausgeschnitten und verarbeitet. Da das Prinzip aber bei allen Funktionen gleich ist, dauert die Gewöhnung nicht allzu lange. Weitere Pinsel-Funktionen ermöglichen das Ausschneiden von ovalen Pinseln sowie die Konstruktion von einfarbigen geometrischen Pinseln (Kreise, Ovale, Rechtecke, Quadrate, gefüllt oder als Rahmen). Ist ein Pinsel einmal ausgeschnitten, können damit noch einige Bearbeitungen, die sich im „Modify Brush“-Menü finden, durchgeführt werden: Pinsel können transparent, als Block oder Silhouette verwendet werden, man kann sie spiegeln oder in jeder Richtung in der Größe verdoppeln oder halbieren oder sie verdrehen. Die „Assimilate“-Funktion schließlich beeinflußt die Farben: Das Bild, in dem ein Pinsel ausgeschnitten wurde, muß ja nicht die gleiche Farbpalette wie das Bestimmungsbild verwenden. Um unter Umständen völlig falsche Pinselfarben zu vermeiden, sucht diese Funktion aus der Palette des Bestimmungsbildes diejenigen Farben heraus, die den Original-Pinselfarben am ähnlichsten sind und koloriert damit den Pinsel neu. Somit hat man immer einen optimal an die neue Farbpalette angepassten Pinsel.

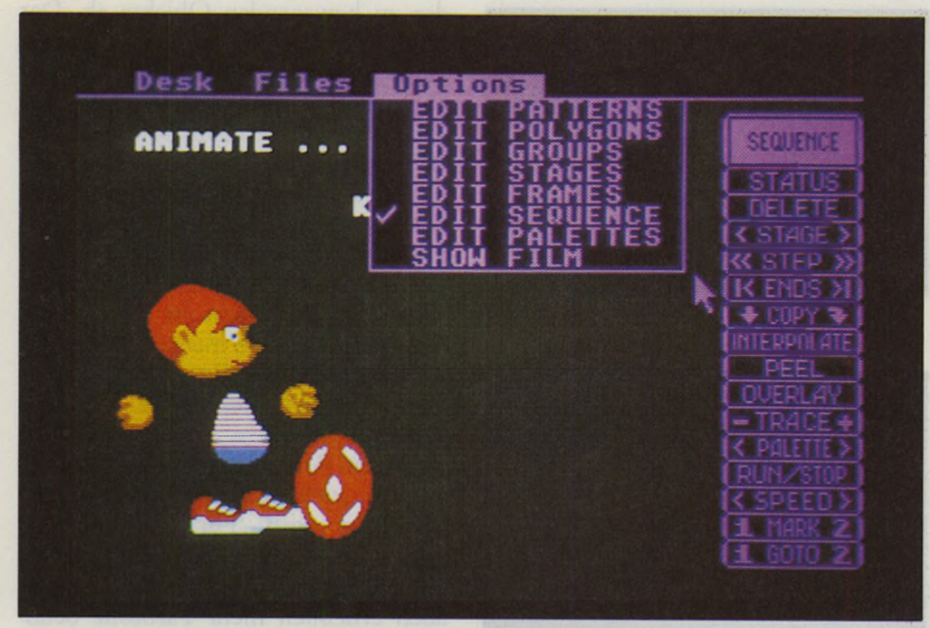


Einige digitalisierte Bilder von Fotohistorischem Wert



Die Toolbox des Pattern-Editors

Das Menü „Line“ erlaubt die Anwahl verschiedener Linientypen. Hier findet man die Standardformen Gerade, aneinanderhängende Geraden, Strahlen, Rechtecke, Kreise, Ovale und Polygone.



Das Options-Menü und die Toolbox des Sequence-Editors

Im „Font"-Menü kann man die gewünschte Fontgröße auswählen, was zu skurrilen Ergebnissen führt, wenn kein Font in der anderen Bildseite enthalten ist.

Dem Laden und Speichern von fertiggestellten Werken dient das „File"-Menü. Sehr komfortabel ist die Möglichkeit, Files zu löschen, oder, falls man überhaupt keinen Platz mehr findet, Disketten zu formatieren.

Ein weiteres Menü nennt sich „Tool". Hier findet man lauter Funktionen, die die Zeichenmöglichkeiten erweitern. Da wäre zuerst einmal eine Möglichkeit, Farben zu verschmieren. Damit kann, wie mit Öl- oder Wasserfarben, eine Linie oder irgend etwas anderes gemaltes mit dem Hintergrund verschmieren. Das „Scrape"-Kommando dient dazu, die Farbe eines Punktes auf der einen Bildschirmseite durch die des gleichen Punktes auf der anderen Seite zu ersetzen. „Melt" mischt die Farben innerhalb eines kleinen rechteckigen Cursors zufällig. Eine „Shade"-Funktion erlaubt es, kleine Bildschirmausschnitte mit der nächsthelleren oder nächstdunkleren Farbe umzufärben. Löcher zwischen Punkten einer bestimmten Farbe können mit den „Patch"-Funktionen auf verschiedene Arten gefüllt werden. „Xcolour" tauscht auf dem ganzen Bildschirm die Farbe von Punkten einer bestimmten Farbe aus (zum Beispiel um alle roten Karosserie-Teile eines Autos blau zu färben). „Outline" umgibt alle Punkte einer bestimmten Farbe mit einer Umrandung in einer anderen Farbe. Etwas ganz besonders feines ist schließlich die „Round off"-Funktion. Bei der Vergrößerung von Bildteilen entstehen meist häßliche Ecken. Die „Round off"-Funktion kann solche Vergrößerungsfehler ausgleichen; aus Ecken werden Schrägen. Die Stärke der Angleichung ist einstellbar. Nach der Vergrößerung sehen Bildausschnitte meist viel besser aus.

Alle diese Funktionen lassen sich mit Hilfe des bereits erwähnten Fensters auf einen bestimmten Bildschirmteil beschränken. Besonders bei der „Round off"-Funktion ist dies notwendig, weil sich in einem Bild ja auch erwünschte Kanten befinden können. Abgerundet werden die Features des Art-Directors durch die Funktionen des „Easel"-Menüs. Hier kann man eine Bildschirmseite auf die andere kopieren oder nur die Farbpalette des anderen Bildschirms übernehmen. Alle Funktionen zur Definition von Arbeitsfenstern und -Rastern, dem Fluchtpunkt für die Perspektive und der Druckgröße finden sich hier. Auch das Drucken einer oder beider Bildschirmseiten wird von hier aus bewerkstelligt. Man kann beide Seiten zusammen in verkleinerter Darstellung betrachten und die Farbrotation an- und ausschalten. Zum Schluß findet man hier noch etwas besonders: Mit der „Def Sprite"-Funktion kann ein beliebiger ovaler oder runder Bildschirmteil ausgeschnitten und mit „Use Sprite" in eine hüpfende Bewegung über den Bildschirm versetzt werden. Dabei kann man Geschwindigkeit und Richtung des Balles frei einstellen. Damit steht der eigenen Amiga-Demo nichts mehr im Wege!

Der Art-Director überzeugt wirklich. Von allen Farbmalprogrammen, die bisher für den Atari vorgestellt wurden, ist er mit Abstand das leistungsfähigste. Nur Deluxe Paint auf dem Amiga läßt sich damit vergleichen. Manche der Features des Programms werden nicht einmal von allen professionellen Video-Grafiksystemen geboten. Nach kurzer Eingewöhnung in das Bedienungsprinzip des Programms ist es auch sehr komfortabel zu bedienen. Ohne Handbuch läuft allerdings zunächst nichts. Das ist auch der einzige Kritikpunkt: Das Handbuch ist zwar ausführlich genug (82 Seiten!), so schnell wie möglich sollten aber ein deutsches Handbuch und eine deutsche Tastaturanpassung erscheinen. Dann könnte man fast wunschlos glücklich sein...

Der Film-Director: Kino auf dem ST

Der Film-Director ist ein Programm, mit dem man auf dem ST Trickfilme erstellen kann. Das Material dafür sind Bilder, die man am besten mit dem Art-Director erstellt. Das Programm arbeitet auch mit Neochrome oder DEGAS zusammen, wenn man die Bildformate mit einem mitgelieferten Programm zuvor konvertiert.

Maximal zwei Bildseiten des Art-Directors (oder eines anderen Malprogramms) sind zunächst das Material. Man beginnt die Dreharbeiten damit, daß man die beiden Bilder mit den Schauspielern und Dekorationen als Bild A oder Bild B in den Film Director hineinlädt.

Als erster Schritt werden aus diesen Bildern die entsprechenden Teile, sogenannte Pattern, herausgeschnitten. Hat man alle Einzelteile auseinandergeschnitten, kann man Teile zu Gruppen zusammenfassen. Das ist sehr nützlich und vereinfacht die Arbeit enorm. Man muß sich nur vorstellen, aus wievielen Einzelteilen ein Trickfilm-Männchen besteht. Für jede Bewegung müssen alle diese Einzelteile in eine bestimmte Lage zueinander gebracht werden. Dafür gibt es zwei

Möglichkeiten: Man könnte alle einzelnen Bewegungsphasen der Figur komplett neu zeichnen. Bei einer etwas größeren Figur wird es dann aber sehr eng auf den beiden verfügbaren Seiten. Die zweite Möglichkeit ist einfacher: Man malt einfach nur die Einzelteile der Figur in allen notwendigen Bewegungsphasen. Dafür braucht man weniger Platz und Zeit. Der Zeitvorteil wäre allerdings gering, wollte man im Film für jedes Bild wieder alle Teile zusammensetzen. Deshalb kann man im Film-Director die Einzelteile eines Objektes zu Gruppen zusammensetzen. So kann man jede Bewegungsphase im Baukastensystem zusammensetzen. und muß nur noch die die Gruppe, die die gerade benötigte Phase enthält, in das Bild einsetzen. Das geht recht schnell, bedeutet aber immer noch genug Arbeit. Ein anderes Beispiel: Der Hintergrund des Filmes soll aus Bäumen bestehen. Um möglichst viele verschiedene Bäume zu bekommen, malt man nur einige Stämme und einige Kronen oder Büsche. Aus diesen Bausteinen setzt man jetzt Bäume oder Baumgruppen zusammen, die man dann einfach mehrfach im Hintergrund abstellt. Einfach ist das auch deshalb, weil alle Plazierungsarbeit mit Hilfe der Maus geschieht, wobei man sich als Positionierungshilfe die Mauskoordinaten einblenden lassen kann.

Außer den Einzelteilen, die man mit den Bildern einlädt, gibt es im Film-Director noch eine zweite Möglichkeit, Objekte zu erzeugen: Eine Funktion zur Erstellung von Linienzügen (Polygonen). Damit kann man einfache geometrische Formen, zum Beispiel die Linien eines Tennisplatzes, ohne große Mühe selbst zu den übrigen Teilen dazumalen. Leider sind diese Linienzüge nicht füllbar, es sind immer nur weiße oder farbige Linien. Sie sind genauso wie die Pattern in Gruppen verwendbar.

Wie geht es nun weiter? Die einzelnen Bewegungsphasen von Schauspielern und Accessoires sind fertiggestellt. Jetzt müssen wir aus diesen Gruppen noch einen Film zusammenstellen. Ein Film besteht aus einzelnen Bildern, englisch „Frames". Im „Frame"-Editor können die zuvor erzeugten Gruppen zu Bildern zusammengesetzt werden. Diese Folge von Bildern ergibt dann den Film. Doch die Schauspieler spielen noch vor einem schwarzen Hintergrund. Also bauen







Die Toolbox des Frame-Editors

wir ihnen eine Bühne: Das funktioniert wie das Erstellen von Gruppen. Einfach Pattern oder Polygone nehmen und auf der Bühne plazieren. Für jeden Frame läßt sich jetzt eine der Bühnen definieren. Damit kann man auch bewegte Hintergründe erzeugen, zum Beispiel eine Häuserreihe, an der eine Figur entlanggeht. Während die Figur geht, wird die Bühne immer ein STück „weitergeschoben".

Alles hat seine Grenzen: Kein Frame darf mehr als 64 bewegte oder unbewegte Bildelemente enthalten. Eine weitere Einschränkung ist die maximale Anzahl der Frames: Maximal 2000 pro Film. Zum Vergleich: Ein Kinofilm besteht aus 24 Frames pro Sekunde. Ein Fernsehbild besteht in Europa aus 25, in Amerika sogar aus 30 Frames. Diese Geschwindigkeit ist notwendig, um dem Auge eine einigermaßen ruckfreie Bewegung und ein halbwegs flimmerfreies Bild zu bieten. Beim Film-Director hängt dagegen die Ablaufgeschwindigkeit von der Menge der zu bewegenden Objekte ab. Der beim Start des Programmes automatisch mitgelieferte Demofilm besteht zum Beispiel aus 1046 Frames und dauert fast zweieinhalb Minuten. Die Bewegungen der Trickfigur sind überraschend fließend. Ich bin gespannt, wie schnell Animation mit dem Film-Director und dem Blitter werden. Das Tempo des Films ist ein neun Stufen einstellbar. Insgesamt ist das Animationstempo für die meisten Filme, die im Rahmen dieses Programms liegen, ausreichend. Professionelle Resultate sind mit Rechnern wie dem ST oder auch dem Amiga ohnehin nicht zu erreichen, auch wenn das immer wieder behauptet wird. Für nur halbwegs professionelle Anwendungen werden sowohl eine höhere Auflösung als auch erheblich mehr Farbtöne benötigt. Bleibt man mit seinen Ansprüchen aber im von den technischen Grenzen gegebenen Rahmen, so wird man mit den Resultaten sehr zufrieden sein.

Über die beschriebenen Besonderheiten hinaus besitzen alle Editoren auch Kopier- und Löschfunktionen. Und das gibt es noch:

Pattern-Editor
In den Pattern-Editor ist eine einfache Editiermöglichkeit für die geladenen Bilder integriert. Dabei handelt es sich um einen einfachen Zeichenstift und eine Lupe. Das ganze ist wohl vor allem zur Ausbesserung von Details gedacht. Veränderte Bilder können abgespeichert werden.

Polygon-Editor
Der Polygon-Editor besitzt ein Kommando zur Erzeugung von Zwischenstufen zwischen zwei Polygonen. Definiert man etwa ein Quadrat und einen Stern, so kann man sich bis zu 99 Zwischenformen berechnen lassen. Läßt man die Zwischenstufen von einem Polygon und einem leeren Schirm berechnen, kann man in der Ferne verschwindende Polygone erzeugen, wenn man die berechnete Folge von Polygonen in aufeinanderfolgende Frames einsetzt.

Group-Editor
Auch hier existiert eine Funktion zur Berechnung von Zwischenphasen von verschiedenen Gruppen. Außerdem können die einzelnen Pattern einer Gruppe horizontal und vertikal vergrößert sowie gespiegelt werden. Mit einem „Layer"-Kommando können Pat-

tern auf verschiedene Ebenen gelegt werden, so daß man bei Überlagerungen immer bestimmen kann, welches Pattern welches verdeckt.

Stage-Editor
Der Bühnen-Editor enthält praktisch die gleichen Funktonen wie der Group-Editor. Auch hier ist also die Berechnung von Zwischenstufen möglich. Darüber hinaus können Texte (zum Beispiel Titel) eingegeben und positioniert werden. Für diese Texte wird der Standard Schrifttyp verwendet. Die Farbe ist wählbar.

Frame-Editor
Auch hier sind im wesentlichen die gleichen Funktionen enthalten. Bei der Zwischenstufenfunktion dürfen auf der gleichen Ebene keine verschiedenen Objekttypen vorhanden sein. Man kann nicht den Übergang eines Kopfes in ein Polygon berechnen lassen, sondern nur z. B. den eines Kopfes in einen anderen. Dabei werden die Pattern verwendet, die sich in der Reihenfolge des Ausschneidens zwischen dem Anfangs- und Endzustand befinden. Für jeden Frame ist eine von acht Farbpaleten auswählbar, die mit einem Farbeditor auch editiert werden kann. Damit können auch Farbrotationseffekte erzeugt werden.
Außerdem ist es möglich, verschiedene Soundfiles mit neun verschiedenen Klangeffekten zu laden. Leider kann man keine eigenen Klangeffekte oder Melodien erzeugen. In jedem Frame kann man einen dieser Effekte beginnen lassen oder unterbrechen.

Sequence-Editor
Mit dem Sequence-Editor können Teile eines bereits fertigen Filmes weiter bearbeitet, zum Beispiel kopiert werden. Auch hier existiert eine Zwischenstufen-Funktion, die allerdings nur die oberste Ebene betrifft und unter Umständen auch den Ablauf des Films beeinträchtigt. Deshalb gibt es auch eine Rücknahme-Möglichkeit. Ein weiteres Kommando verdoppelt die oberste Ebene eines Filmabschnitts. Damit könnte man zum Beispiel aus wenigen fliegenden Vögeln eine große Menge machen.

Mit diesen Möglichkeiten erfordert der Film-Editor eine lange Einarbeitungszeit. Dabei wird man vom Handbuch, das nur knapp 50 Seiten umfaßt, nur unzureichend unterstützt. Überhaupt ist das Handbuch eine Katastrophe: Im Gegensatz zum Art-Director wird hier die Einarbeitung zum Abenteuer. Daß ein Progrmm von diesem Schwierigkeitsgrad nicht in fünf Minuten vollständig zu verstehen ist, ist selbstverständlich. Aber mit diesem Handbuch verliert man schnell die Lust an der Arbeit. Das liegt nicht nur an der Kürze, sondern auch an dem miserablen Englisch, in dem es geschrieben ist.

Zusammenfassung: Der Film-Director ist ein mächtiges Programm. Die kreative Vorarbeit, das Malen der Szenerie und die genaue Planung des Ablaufs nimmt er dem Trickfilmregisseur natürlich nicht ab. Aber er erleichtert viele Arbeiten. Vor allem zusammen mit dem Art-Director kann man dem Computergrafikbegeisterten, dessen Etat für die Anschaffung einer VAX nicht ausreicht, den Film-Director nur empfehlen. Die Bedienung ist zwar gewöhnungsbedürftig, aber dies ist wohl der Komplexität des Problems zuzuschreiben. Gespannt darf man sein, ob sich der vom Amiga her bekannte AEGIS-Animator in seiner Atari-Version mit dem Direktorenteam messen kann. Außerdem wird es spannend, wenn für den Atari erst ein Videorecorder-Anschluß erhältlich ist. Welche Möglichkeiten sich mit einem Farbdigitizer erst ergeben!

Fazit: Prädikat Besonders wertvoll.

Christian Schormann

Art Director
preis: 209,– DM

Film Director
Preis: 235,– DM

von Andromeda Soft, Vertrieb PDS

Deutscher Vertrieb:
Application Systems III Heidelberg
Brückenstr. 47
6900 Heidelberg
Tel. 0 62 21 / 41 01 34

Computer-Technik Kieckbusch
Am Seeufer 11+22
5412 Ransbach
Tel. 0 26 23 / 16 18

TEMPUS
oder
Wie GEM das Laufen lernte

Das meistverwendete Programm jedes Programmierers ist der Editor. Besonders bei Compilersprachen ist es wichtig, den Quellcode als Text eingeben und abspeichern zu können. Mit dem Anspruch, bessere und schnellere Sprachen zu entwickeln, wuchs auch das Verlangen nach besseren Editoren. Um so verwunderlicher ist es, daß erst jetzt ein Editor entwickelt wurde, der keine Wünsche offen läßt. Wir haben die endgültige Version dieses 'Werkzeuges' unter die Lupe genommen.

Was zuerst auffällt: Der Editor wurde unter GEM geschrieben, was der Bedienerfreundlichkeit zugute kommt. Da unter GEM geschriebene Editoren meist zu langsam waren, ließen einige Firmen ihre Editoren nur unter TOS laufen. Daß GEM keineswegs langsam sein muß, beweist TEMPUS.

Um die Geschwindigkeit des Programms zu testen, haben wir eine Datei geladen, die ca. 70 Kilobytes groß ist. Die Ladezeit und einige andere Merkmale haben wir mit anderen Editoren verglichen, so daß Sie selbst vergleichen können. Die Ergebnisse finden sich in Tabelle 1.

Editor-Benchmarks

	1st_Word	MCC-Ed	GST-Editor	Ideal	Tempus
Laden einer 70 KByte Datei	37 s	39 s	37 s	6 s	10 s
Suchen nach Wort am Textende	25 s	5 s	21 s	2 s	★
Ersetzen aller „e" durch „ae"	420 s	1088 s	79 s	514 s	15 s

★ war bei Tempus nicht meßbar (ca. 0,5 s)

Tabelle 1

Bild 2: „Suchen & Ersetzen leicht gemacht"

Ein Meister der Geschwindigkeit

Sein wahres Gesicht zeigt TEMPUS allerdings erst, wenn man beginnt, Texte zu editieren und zu bearbeiten. Gibt man häufig Texte ein, so fällt auf, daß, wenn der Cursor zu schnell eingestellt ist und beim Scrollen die Tastenwiederholfunktion benutzt wird, der Text nach dem Loslassen der Taste meist noch einige Zeilen weiterläuft. TEMPUS dagegen verschiebt den Bildschirminhalt so erstaunlich schnell nach oben oder unten, daß bei normaler Cursoreinstellung kein und bei schneller Cursorgeschwindigkeit nur ein minimales Nachlaufen zu bemerken ist – und das, obwohl dieser Editor nicht unter TOS, sondern unter GEM läuft. Als die Redaktion TEMPUS zum ersten Mal scrollen sah, waren alle stark beeindruckt. Auch hier haben wir einige Vergleiche mit anderen Editoren gemacht. Während man bei 1st_Word fast zuschauen kann, wie eine Zeile mehr ruckartig auf dem Bildschirm erscheint, und die Geschwindigkeit im MEGAMAX-Editor gerade noch erträglich ist, huscht bei TEMPUS der Text wie ein Schleier über den Bildschirm.

Um die Geschwindigkeit zu realisieren, wurde der Editor vollständig in Assembler geschrieben. Doch nicht nur die fantastische Scrollgeschwindigkeit, auch alle anderen Funktionen können sich sehen lassen. So erledigt TEMPUS zum Beispiel bis zu 562 (!) Vertauschungen pro Sekunde. Wenn

```
Text
-----Text laden------
   neuladen...
   Überladen
   hinzuladen
---Text speichern----
 ... mit Backup
 ... ohne Backup
 ... unter...
---Block speichern---
 ... unter...
---------------------
 Tempus verlassen

Marken
---Marke anspringen---
 ◇ Marke >A<  >.....
 ◇ Marke >B<  >....
 ◇ Marke >C<  >....
 ◇ Marke >D<
 ◇ Marke >E<
------setzen von------
 ... Marke >A<
 ... Marke >B<
 ... Marke >C<
 ... Marke >D<
 ... Marke >E<

Suchen
 Zeichenkette suchen...
 Suchen wiederholen
 Suchen & Ersetzen...
---Position anspringen---
 ◇ Zeile...
 ◇ Seite...
 ◇ Textanfang
 ◇ Textende          >....S6
 ◇ Blockanfang
 ◇ Blockende
 ◇ letzten Pos.

Block
----Block-Marken-----
 Start festlegen
 Ende festsetzen

 Markenkomplement
 Marken löschen
--Block-Operationen--

 Block verschieben

 Block kopieren

 Block löschen

Drucken
-----Ausdrucken--------
 ... gesamten Text
 ... Blockbereich
-----Installationen----
 ... übernehmen
 ... sichern auf Disk

Speziell
---Tabulatorzeichen---
 Text expandieren
 Text komprimieren
----Zeichensatz-------
 ... 8*16er laden...
 ...  8*8er laden...
 System-Zeichensatz
----Fensterlage-------
 ... überlappend
 ... nebeneinander
 ... untereinander
----------------------
 Querverweisliste...
 UPN-Rechner
 Funktionstasten...

Modus
 Einfügemodus
 Überschreibmodus
 Autom. Einrücken
---Zeichengröße------
 ... 8*16-Matrix
 ...  8*8-Matrix
--Zeichendarstellung--
 ... invers
----------------------
 Zeilenlänge 160
 Seitenlänge  72
 Umblättern    + 0
 Tabulatorweite   8
 Tabulatorweite 5
 Tabulatorweite 8

Hilfen
 Text
 Marken
 Suchen
 Block
 Drucken
 Speziell
 Modus
```

Bild 1: „Die Menüleiste von Tempus"

er allerdings gegen längere Strings austauschen muß, wird er etwas langsamer – dann schafft er 'nur' noch 500 Vertauschungen. Auch hier haben wir einen Vergleichstest gemacht: In unserem 70k langen Text kam 7440 mal der Buchstabe 'e' vor. Diesen haben wir durch 'ae' austauschen lassen, das heißt, wir haben Tempus mit der schwierigeren Arbeit beauftragt. Der Metacomco-Editor hat für diese Arbeit am längsten gebraucht: Er brachte es auf ganze 1088 Sekunden – das sind über 18 Minuten! 1st_Word schlug sich tapfer mit 7 Minuten. Als wir den Test mit Tempus durchführten, trauten wir unseren Augen nicht: Als wolle er die anderen Editoren ärgern, meldete er sich nach sage und schreibe 15,2 Sekunden mit der Meldung, er habe gerade 7440 Stellen ersetzt. Zudem ist die Austauschfunktion nicht nur schnell, sondern auch äußerst komfortabel (Bild 2).

Ähnlich überragend stellte sich TEMPUS auch in der Suchfunktion dar: Beim Suchen nach einem Wort, das am Ende des Textes stand, brauchte 1st_Word 25 und der Metacomco-Editor 5 Sekunden. Die Zeit von TEMPUS war nicht feststellbar; sie dürfte etwa bei einer halben Sekunde gelegen haben.

CCD liefert mit TEMPUS indes nicht nur einen schnellen, sondern auch einen komfortablen Editor. Die Anzahl der Funktionen ist kaum zu überbieten. Bevor auf die interessantesten Funktionen eingegangen wird, sollten Sie sich einmal Bild 1 und Tabelle 2 anschauen. Die Vielfalt der Menüleiste wird nur noch von der Menge der Befehle übertroffen. Die in Tabelle 2 dargestellten Befehle sind **alle** über die Tastatur verfügbar. Dabei wurden, soweit es ging, sogar Ähnlichkeiten mit WORDSTAR-Codes hergestellt, um eine schnellere Einarbeitungszeit zu gewährleisten. Selbst GEM-Gegner kommen durch die Tastenbelegung aller Befehle auf ihre Kosten. Zusätzlich sind die Funktionstasten mit Texten belegbar. Wer also bestimmte Wörter (oder auch ganze Sätze) öfters benötigt, legt sie auf den Funktionstasten ab und kann sie dann mit einem Tastendruck wieder aufrufen.

Befehlsübersicht von TEMPUS – alle erreichbar über Tastatur:

zum Textanfang	Blockstart festlegen
zum Textende	Blockende festlegen
zur letzten Position	Block speichern unter...
zur Zeile...	Block verschieben
zur Seite...	Block kopieren
Wort nach links	Block löschen
Wort nach rechts	Markenkomplement
Bildschirmseite abwärts	Blockmarken löschen
Bildschirmseite aufwärts	Blockbereich drucken
nächste Tabulatorposition anspringen	zum Blockanfang
Zeile löschen	zum Blockende
Wort unter Zeiger ab Position löschen	Text überladen
Zeiger in obere linke Fensterecke	Text speichern m. Backup
Zeiger zum Textanfang	Text speichern o. Backup
Marke A bis E setzen	Text speichern unter...
Marke A bis E anspringen	Text zuladen
8x16er Zeichensatz laden	Fensterlage überlappend
8x8er Zeichensatz laden	Fensterlage nebeneinander
Text neuladen	Fensterlage untereinander
Text drucken	neue Zeilenlänge
Installation sichern auf Disk	neue Seitenlänge
Zeichenkette suchen	Umblätterfaktor
Suchen und Ersetzen	neue Tabulatorweite
Suchen wiederholen	Tabulatorweite 5
Querverweisliste	Tabulatorweite 8
System-Zeichensatz	Automatisches Einrücken
8x8 Zeichensatz	Überschreibmodus
16x8 Zeichensatz	Einfügemodus
Inversdarstellung	Speicher
Funktionstastendefinition	Installation übernehmen
Zeichenauswahl bzw. Tabelle	Infobox
Speicher reorganisieren	TEMPUS verlassen

Tabelle 2

GEM wurde ein wenig 'überarbeitet

Nach dem Laden von Tempus fällt zunächst die neue Fileselektorbox auf (Bild 3). Abgesehen davon, daß die

Originalbox beim Eingeben eines Unterstrichs abstürzt, läßt sie manches zu wünschen übrig. Diesen Gedankengang muß man auch bei CCD gehabt haben, denn in Tempus findet sich eine völlig neue Dateiauswahlbox. Nicht nur, daß man nun endlich sieht, ob man einen Text lädt oder abspeichert, sie bietet auch sonst einige Features:

1. Die Diskettenlaufwerke sind anklickbar. Dabei werden nur die Laufwerke angezeigt, die auch wirklich zur Verfügung stehen.
2. Der Pfadname darf 42 Zeichen lang sein.
3. Die gebräuchlichsten Extender ("★.★", "★.pas" etc.) sind durch Anklicken wählbar.
4. Die einzelnen Dateiattribute wie Filelänge, Datum, Uhrzeit sind abrufbar.

Eine tolle Sache, wenn man bedenkt, daß man normalerweise wegen solcher Fragen das DESKTOP konsultieren muß.

Eine zweite Besonderheit: Tempus enthält einen Taschenrechner, mit dem man nebenbei kleinere Rechnungen durchführen kann. Er entspricht den üblichen Taschenrechnern, mit dem Unterschied, daß er Dezimal, Hexadezimal und Binär rechnen kann. Dabei wurden auch logische Operationen einbezogen.

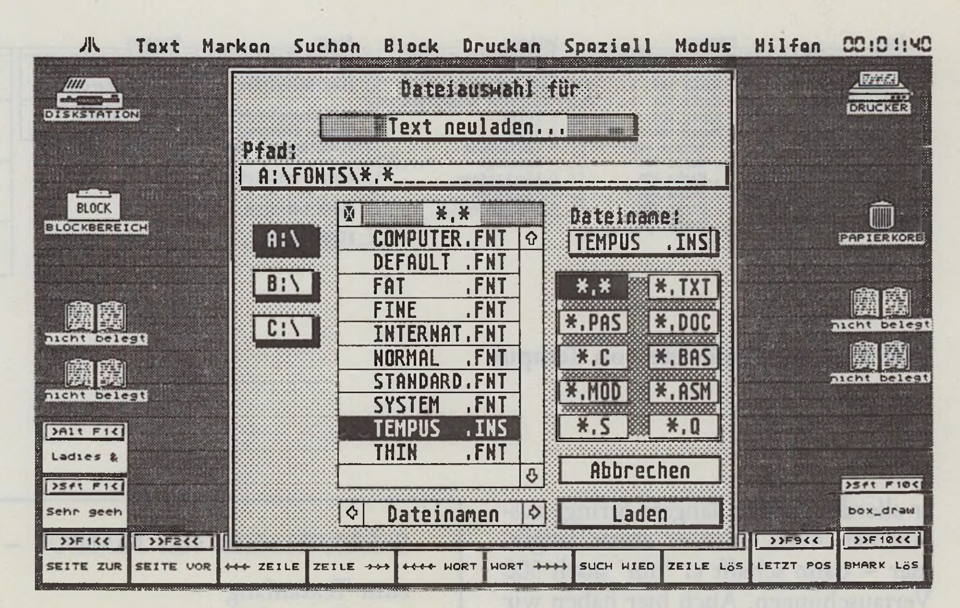


Bild 3: „Die neue Fileselektorbox"

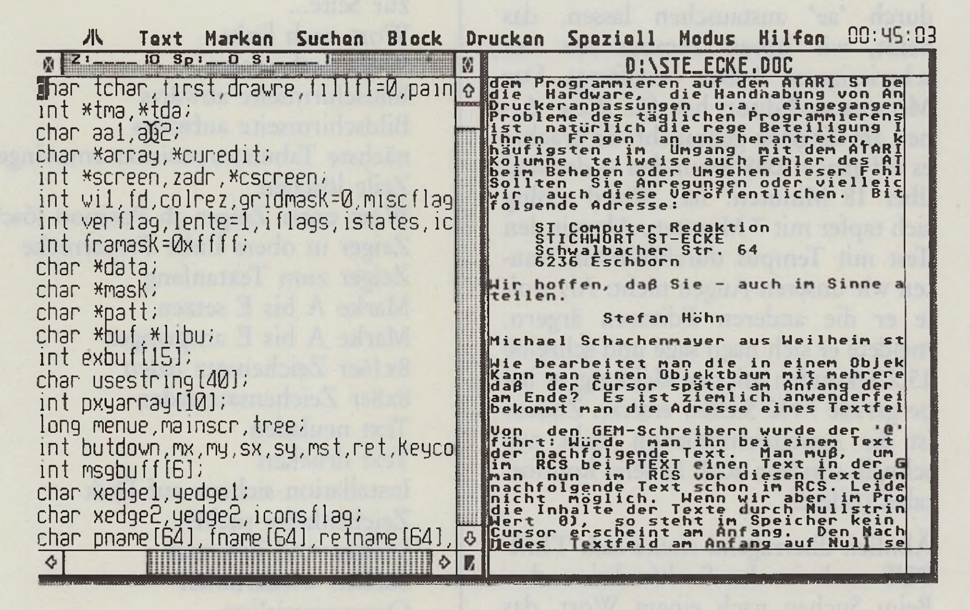


Bild 4: „Verschiedene Zeichensätze gleichzeitig"

Querverweisliste inklusive

Man kann sogar eine Querverweisliste (Cross-Reference) anlegen, mit der sich (natürlich sekundenschnell) alle Stellen auffinden lassen, an denen zum Beispiel eine bestimmte Variable steht. Klickt man dann in dem Fenster, in dem sich die Querverweisliste befindet, die entsprechende Zeilennummer an, so springt der Cursor im gerade bearbeiteten Text automatisch in diese Zeile. Eine neue und sehr sinnvolle Funktion, zumal dieses Springen in eine bestimmte Zeile nicht nur mit der Querverweisliste, sondern auch mit anderen Texten funktioniert. Hatte man zum Beispiel beim Compilieren mit MEGAMAX einen Fehler, so lädt TEMPUS den Quellcode **und** die Fehlerdatei. Klickt man die Zeilennummer der Fehlerdatei an, springt man sofort in die fehlerhafte Zeile. Eine tolle Idee! Sollte dem Anwender der eingebaute Zeichensatz nicht zusagen, so kann er Zeichensätze im DEGAS-Format laden. Intereessant ist, daß man bei den bis zu vier **gleichzeitig** bearbeitbaren Texten **jeweils** auswählen kann, ob man einen 8x8- oder 16x8-Font benutzen kann. Hilfreich ist, daß Tempus auf Kommando die Fenster der gerade in Arbeit befindlichen Texte wahlweise nebeneinander oder untereinander stellen kann. So kann man sehr gut Texte vergleichen und bearbeiten.

Installationen abspeicherbar

Selbst an eine Druckeranpassung wurde gedacht. Dadurch kann sich jeder eine Anpassung für seinen Drucker erstellen und wird ab sofort keine Probleme mehr haben, Texte oder markierte Textstellen (Blöcke) auszudrucken. Man kann die Icons wie Drucker, Mülleimer, Diskstation etc. nach Belieben umherschieben und so postieren, wie man es für sinnvolll hält. Daß all diese Einstellungen (selbst Tabulator und Zeilenlänge werden be-

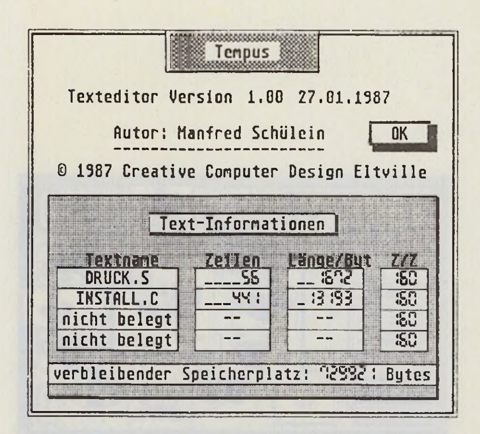


Bild 5: „Alle wichtigen Daten auf einen Blick"

rücksichtigt) abspeicherbar sind, erscheint bei diesem Programm fast selbstverständlich – TEMPUS lädt Sie dann beim Starten selbständig.

Resumee

Man könnte noch einige Feinheiten und Neuigkeiten ansprechen, sie würden allerdings den Rahmen unseres Berichtes sprengen. Alles in allem kann man diesen Editor als ein gelungenes Werk bezeichnen. Nicht nur, daß er durch die Assemblerprogrammierung konkurrenzlos schnell ist und Speicherplatz spart, Tempus wurde auch an alle verbreiteten Shells angepaßt – eine frohe Nachricht für diejenigen, die sich mit den mitgelieferten Editoren der entsprechenden Entwicklungssysteme (zum Beispiel TOS-Editoren oder langsame GEM-Editoren) herumärgern mußten. Zugegeben, der Test klingt sehr euphorisch – doch es ist äußerst wahrscheinlich, daß TEMPUS neue Maßstäbe setzen wird. Sollten Sie sich nicht sicher sein, ob Ihnen dieser Editor zusagt: Bei CCD it eine Demo-Version erhältlich. TEMPUS ist besonders für alle diejenigen ein Muß und eine Bereicherung, die sehr gern angenehm und schnell arbeiten. Wer möchte das nicht? Auch der Preis von 79 DM (!) läßt vermuten, daß Tempus sehr bald in aller Munde sein wird...

Stefan Höhn / (HE)

Bezugsquelle:
CCD – D. Beyelstein
Burgstraße 9
D-6228 Eltville

Neu Neu Neu

Ab sofort haben wir unseren Public-Domain-Service umgestellt. Wegen der großen Zahl von Disketten und dem großen Aufwand bei der bisherigen Versandart haben wir uns entschlossen, die Public-Domain-Software mitsamt den Disketten zu liefern. In Zukunft bieten wir die Public-Domain-Software auf sehr guten Disketten zum Unkostenbeitrag von je DM 10,– an.
Bei den Disketten handelt es sich nicht um Noname-Produkte, sondern um qualitativ hochwertige Disketten der Marke **FUJI FILM MF1DD (einseitig 135 TPI)**, mit denen wir gute Erfahrungen sammelten.

Diese Regelung bietet folgende Vorteile:
- mühevolle Verpackung, Formatieren und Beschriften der Disketten, Hin- und Herschicken entfällt.
- einfachere Bearbeitung bei Ihnen und bei uns.
- kürzere Liferzeiten.
- Sie müssen sich nicht um Verpackungsmaterial kümmern, das übernehmen wir.
- und Disketten braucht schließlich jeder, noch dazu zu diesem Preis.

Beachten Sie bitte die Versandbedingungen am Ende der folgenden Seite.

Momentan sind folgende Public-Domain-Disketten vorhanden:

1. Diskette
● *Neochrome V.06:* Malprogramm [f] ● *Doodle:* Malprogramm [s/w]

2. Diskette
● *Joshua:* Diskmonitor – CP/M Emulator (CP/M 80) ● *Megaroids:* Weltraumspiel ● *Diskformatter:* erweitert Diskkapazität auf 399 K bzw. 809 K ● *Kartei-Kasten:* luxuriöses Karteiprogramm [s/w] ● *50/60 Hz*-Umschaltung für Monitore.

Diskette 3 und 4 beinhalten ein komplettes *FORTH*-System *(volksFORTH-83)*. Das System umfaßt einen Screen-Editor, Interpreter, Compiler und diverse Utilities.

5. Diskette
● *Ramdisk* mit Source-Code ● *SG 10* Druckertreiber als Desk-Accessory ● *JAM Term:* Terminalprogramm ● *Diskcopy* ● *ST Grafic:* bewegte Grafiken in 3D ● *CP/M Emulator V 8.4:* neuste Version

6. Diskette
Diese Diskette beinhaltet einige Nutzprogramme für den CP/M-Emulator. Lauffähig mit Version 8.4.
● *CPMTOS:* TOSCPM wandelt CP/M-ASCII-Files in TOS-Format und zurück ● *SCOPY:* Kopierprogramm für ein Laufwerk ● *DDTZ:* Debugger ● *SETKEY:* Tastaturbelegung, u. v. m.!!

7. Diskette
Die Sprache *LISP* (künstliche Intelligenz) ist nun auch auf Public-Domain erhältlich. (incl. 54 kByte Dokumentation, Anleitung und Beispielen)

8. Diskette
● *Diskmon:* ● *GfA Basic-Demo:* ● *Kuvert:* zum Bedrucken von Kuverts ● *Mini-CAD:* ein praktisches Programm mit Source-Code ● *Vokabel:* Vokabeltrainer ● *Superformat:* unterschiedliche Diskformate

9. Diskette
● *Funkplot:* Funktionsplotprogramm incl. Funktionseingabe über Tastatur, Integration und Differenzation ● *Konvert:* Konvertierung von Farb- auf Monochrombilder in 3,6 Sekunden ● *Spriteed:* ein kleiner Spriteeditor ● *Super:* ein Superhirnspiel [f]

10. Diskette
● *Iconeditor* ● *Puzzlepuzzle:* Puzzleprogramm ● *Krabat-Schach* Schach für den ST

11. Diskette
● *PROLOG 10* Die Sprache der neuen Generation incl. 95 KByte Anleitung.

12. Diskette
● *Rechner:* ein luxuriöser Taschenrechner ● *Calc:* Taschenrechner als Desk-Accessory ● *Biorhythmus:* Erläuterung überflüssig

13. Diskette
● *Pascalshell:* GEM-Menüsteuerung für ST-Pascal. ● *MMCOPY .ACC:* Filecopy als Accessoir ● *Happy 4:*'Vier gewinnt' ● *Labyrint. ACC:* Labyrint als Accessoir ● *MAD . ACC:* Ihr Rechner entwickelt Eigenleben ● u. a.

14. Diskette
● *Tempelmon:* Monitor von Th. Tempelmann [s/w] ● *Transistorvergleicher:* Transistordatenverwaltung ● *Bitte ein Bit:* Bitcopyprogramm ● *Diskmon* ● *VIFibu:* Kleine Finanzbuchhaltung ● *EX 06:* Fängt Systemfehler und zeigt sie an.

15. Diskette
Druckeranpassung 1st_Word
Auf dieser Diskette befinden sich alle uns zur Verfügung stehenden Druckertreiber. Es sind folgende Treiber enthalten:

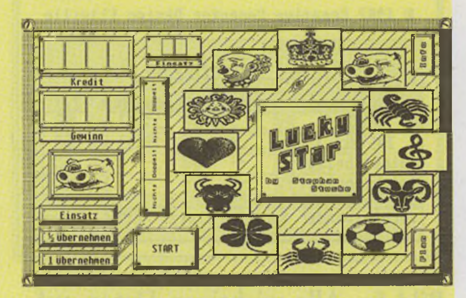
ASCII	CP_80	Epson_12
Epson_LX80	Epson RX80	Gabi 9009
IBM	IBM-Komp	KX-P1092G
LQ_800	MP_165	NEC_P2&P3
NEC_P6&P7	NL10_IBM	NL10_Para
OKI_M182	OKI_M192	QUME
SMM804	TA-SE320	TAXAN 810ST
C 8028	Corona D300	Gabi 9009

16. Diskette
Sammlungen diverser Programme incl. Source Code [s/w] ● *Etikettendruck:* bedruckt Etiketten (Basic) ● *Funkplot:* mit Funktionseingabe (Basic) ● *Kassetten- und Videodatei* unter GEM (Basic) ● *Typewriter:* Schreibmaschinenkurs (Basic) ● *LIFE:* Fortpflanzungs-Rechnung (Pascal) ● *Schiffe:* Schiffe versenken (Demo für GEM-Routinen) (Basic)

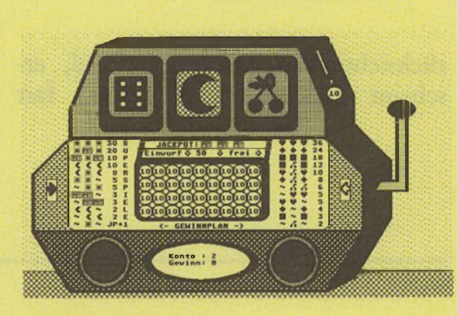
17. Diskette
Programmsammlungsdiskette incl. Source [s/w] ● *Aktien:* Der Aktienmarkt ● *Habu:* Verwaltung von Konten ● *Lohnbuchhaltung & Versanddruck* ● *Steuer:* Einkommens- und Lohnsteuerberechnung.

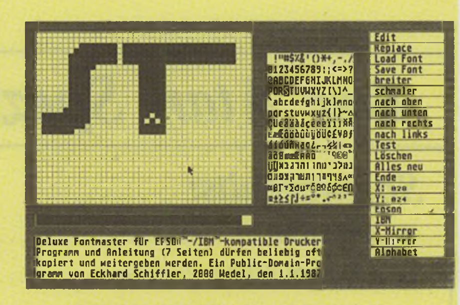
18. Diskette
● *Doodle plus:* erweiterte Doodle Version; incl. Bilder ● *Power:* Sonderzeichensatz [s/w] ● *Konverter:* Wandelt alle Bildformate ● *Datum:* Datum auf Disk ● *Smooth:* Scrollt eine 64 000 KB Doodle-Bild ● *Format:* erzeugt diverse Formate ● *Kbug:* ST leidet unter Käferbefall

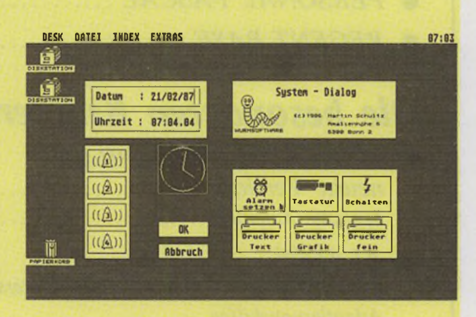
19. Diskette
● *Kermit plus:* Stark verbesserte GEM-Version des Terminalprogramms ● *Disksort:* Druckt Diskinhalt als Liste oder als Etikette ● *Terminal:* Terminalprogramm ● *Fontedit:* Zeichensatzeditor für GEM-Draw und Easy-Draw Fonts. [s/w] ● *OKI-Init.ACC:* Steuercodes für Microline 182 incl. Source-Code



Lucky Star (PD 39)


Jackpot (PD 39)

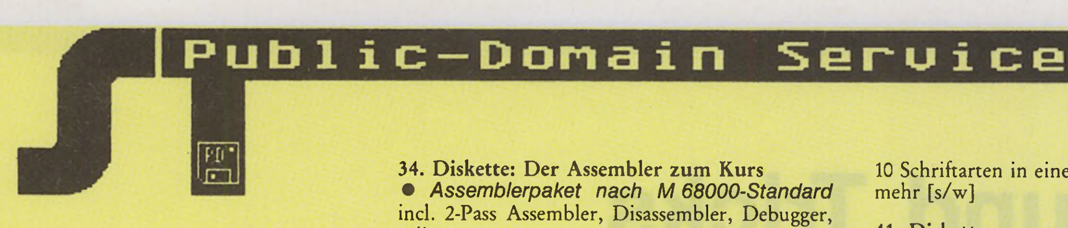

Deluxe Fontmaster (PD 40)



System ACL (PD 43)

20. Diskette
● *Harddisk:* Treiber für Harddisk. Automatischer Start der Platte (keine Bootdiskette nötig) ● *Toram:* Kopiert Programme in eine Ramdisk ● *RSCAnalyse:* Analyse von RCS-Files [s/w] ● *DESK Uhr.ACC:* Zeigt ständig Zeit und Datum an. Drei einstellbare Weckzeiten. ● *GEMTOOL:* Demo für GEM-Programmierung [s/w]

21. Diskette
● *Adress:* Private Adressverwaltung ● *Etikett:* Etikettendruck ● *Haushalt:* Privatkontenverwaltung ● *Selektor:* Dateiverwaltung (30 frei erstellbare Felder)



22. Diskette
● *Plot:* Grafische Darstellung von Wertepaaren ● *Myterm:* Terminalprogramm (GfA-Basic) ● *VKT:* Vokabeltrainer

23. Diskette
● *DGDB:* 'Das große deutsche Ballerspiel', Aktionspiel siehe RELAX 12/86 [f] ● *Kalah:* Das bekannte Brettspiel ● *Fractals:* Schnelle 'Apfelmännchen-Version' in Assembler.

24. Diskette
● *Taurus:* Strategiespiel für mehrere Spieler. RELAX 12/86. Schöne Grafik, viele Optionen [f], (leider nur Mega-Rechner)

25. Diskette
● *Disk Engineer:* Diskutility (Rescratch, ...) ● *Omikron Basic:* Run-Only Interpreter des neuen Basics

26. Diskette
● *Codetabelle:* ASCII-Tabelle (Accessoir) [s/w] ● *Life_Game:* Das bekannten Generationenspiel (Accessoir) [s/w] ● *Term:* Und noch ein Terminalprogramm ● *Biorhythmus:* ST-Basic schlägt zurück [s/w] ● *Konto:* Kontoführung in GfA-Basic [s/w] ● *KX P1091:* Accessoir zur Druckersteuerung

27. Diskette
● *NEC-Treiber:*Hardcopy-Treiber für 24-Nadeldrucker (NEC und kompatible). Aufruf über ALT+HELP ● *ST-Sound Demo:* Die Siegerdemo des ST-Computer-Sound-Wettbewerbs. 'Zottl, Knurp und Schnatter hauen rein' (siehe Titelbild ST-Computer Nr. 12)

28. Diskette
● *GfA .Basic Programme incl. Source* ● *Funkplot:* [s/w] ● *Terminal:* Terminalprogramm in GfA [s/w] ● *Konverter:* Konvertiert ST-Basic in GfA-Basic ● *ST-Kill:* Utilityprogramme ● *Sprite-Editor:* [s/w]

29. Diskette
● *Diskutility:* Der Weg ans Directory (manipuliert Löschschutz, erzeugt waschechtes IBM-Format, und mehr) [incl. Source in Pascal] ● *Adress_pas:* Adressverwaltung, die einige käufliche Programme das Fürchten lehrt. ● *ST-Graph:* 3-D Balkengrafik [s/w] [incl. Source in GfA]

30. Diskette: Spielesammlung
● *Bohnen* ● *Schiffe:* Schiffe versenken
● *Mastermind:* Superhirn für Superhirne [f]
● *Dragon_hunt:* Ein äußerst gefräßiger Drachen will ausgerechnet Ihnen an den Kragen [f]
● *Memo:* Memory (2 Versionen) [s/w]
● *Solitaire:* Das bekannte 'Taschenspiel' [s/w]
● *4 Gewinnt:* Spiel in GfA-Basic incl. Source [s/w]

31. Diskette: Spielesammlung
● *Patience:* Wunderschöne Version dieses Kartenspiels (siehe Spieletest Nr. 1/87 [s/w] ● *3D-Irrgarten:* Finden Sie einen Weg aus einem dreidimensionalen Labyrinth. Trotz ST-Basic sehr schnell [s/w] ● *Avalun:* Sie verwalten ein Königreich. (ST-Basic)

32. Diskette
● *Mathlib:* Eine umfangreiche Mathematikbibliothek für Pascal. Trigonometrische, Matrizenalgebra und vieles mehr, direkt zum Einbinden ● *Finanzmathe:* Berechnung von Zinsen, Laufzeiten, Laufzeit, Tilgungsplänen, etc.

33. Diskette: diverse Utilities
● *Encrypt:* Ver- und Entschlüsseln von Dateien ● *Ramdisk:* Ramdisk beliebiger Größe ● *Recover:* Regeneriert gelöschte Files (!!) ● *Sterm:* VT52 Terminal + Source in Assembler ● *Freeram:* Zeigt freien Speicher

34. Diskette: Der Assembler zum Kurs
● *Assemblerpaket nach M 68000-Standard* incl. 2-Pass Assembler, Disassembler, Debugger, Editor.

35. Diskette
● *Andromeda:* Zeichenprogramm (DEGAS-ähnlich) ZOOM, Beschriften in vier Richtungen und allen Größen, Kopieren, Linienstil / Art / Breite und alle normalen Grafikbefehle [s/w] ● *3D-Konstruktion:* Zum Entwurf dreidimensionaler Gegenstände [s/w] ● *Globus:* Animierte Erdballgrafik

36. Diskette
● *CLI:* vollständiger Kommandointerpreter (CP/M, MS-DOS Oberfläche) ohne Maus. Ausführen von Batch-Files, Autostart, Kopieren auf Ramdisk, Belegung der Funktionstasten, Ausführen von Programmen (TOS + GEM), etc. ● *IBM. Drive:* Ermöglicht den Filetransfer IBM (–) ST ● *Drupatch:* Zum Ausdruck von Source Files. Mit Zeichenwandlung wie 1ST Word, universell anpaßbar, Perforationssprung. ● *DB_Etikett:* Druckt Etiketten aus DB-Master

37. Diskette
● *SCIENTFIC:* Zeichensatzeditor. Erzeugt Anpassungsfile für 1st Word [s/w] ● *Convert:* Konvertiert ST- und Microsoft Basic Programme in GfA-Basic ● *Check:* Rechtschreibkorrekturprogramm. Erstellt eigenes Lexikon, Übernahme der neuen Wörter, Korrektur direkt im Text.

38. Diskette
Alle Bilder des ST-Kalenderwettbewerbs, die in Heft 1/87 veröffentlicht wurden, incl. Picture-Compactor.

39. Diskette
● *Jackpot:* Einarmiger Bandit, incl. Sourcecode in GFA-Basic [s/w] ● *Lucky:* Spielautomat [s/w] ● *Hanoi:* Türme von Hanoi.

40. Diskette
● *Deluxe Fontmaster ST:* Druckprogramm zum Erstellen von Karten und Tabellen, mit vielen Zeichensätzen, Porportionalschrift und Blocksatz, eigener Text- und Zeichensatzeditor. Bis zu 10 Schriftarten in einem Text mischen und vieles mehr [s/w]

41. Diskette
● *Painter:* Zeichenprogramm unter GEM. Verfügt über einige Besonderheiten. Z. B. Zoom, Linien und Füllmustereditor, Blockoperationen (Verschieben, Drehen...) [s/w] ● *VIKA:* Bedrucken von Kassetten und Videohüllen [s/w] ● *Kalender:* Diverse Kalenderberechnungen, u. a. Ausgabe eines Jahreskalenders auf Drucker oder Bildschirm, mit Anzeige aller Feiertage (sehr wichtig).

42. Diskette
● *FRAM 77:* Statistische Berechnungen für Konstrukteure. Sehr umfangreich ● *Meßwert:* Grafische Darstellung von Meßwerten [s/w], incl. Sourcecode in Pascal.

43. Diskette
● *System:* Accessoir ähnlich dem Kontrollfeld. Besonderheiten: Uhr und 4 Weckzeiten, Druckereinstellung (Rand, Schriftart...) [s/w], incl. Sourcecode in Pascal ● *Snaphelp:* Hiermit kann SNAPSHOT (1st Word Utility) auch auf DEGAS angewendet werden, incl. Sourcecode in Pascal.

Sonderdisks

Die folgenden Programme sind nicht Public-Domain. Sie können aber bei uns bezogen werden.

A.) – *TOS:* Die letzte Disketten-Version des TOS vom 6.2.1986. Zum fehlerfreien Betrieb des Rechners benötigen Sie diese Version, da z. B. mit älteren Versionen manche Ordner nicht gelesen werden (0 Bytes free).
Unkostenbeitrag 15,– DM incl. Diskette.

B.) – *RCS:* Das Resource Construction Set aus dem ATARI-Entwicklungspaket. Unverzichtbar bei der Erzeugung von Objektbäumen, Dialogboxen, Drop-Down Menüs. Kurzum: Ein Muß für GEM-Programmierung.
Unkostenbeitrag 15,– DM incl. Diskette.

Bis auf die Höhe des Unkostenbeitrags gelten die gleichen Versandbedingungen wie bei der Public-Domain-Software.

Versandbedingungen:

Um einen schnellen und problemlosen Versand zu gewährleisten, beachten Sie bitte folgende Punkte:

- Legen Sie pro bestellter Diskette DM 10,- als Verrechnungsscheck bei (bei Sonderdisks DM 15,-)
- Fügen Sie dem Betrag folgende Versandkosten (Porto+Verpackung) bei:
  Inland DM 5,– Ausland DM 10,–
- Legen Sie unbedingt einen Aufkleber mit Ihrer vollständigen Adresse bei.
- Für ausländische Besteller sei erwähnt, daß Sie mit Euroschecks auch in DM bezahlen können.
- Bitte legen Sie keine zusätzlichen Bestellungen bei (Abo, Zeitschriften, Platinen, etc.), da dies den Vorgang erheblich verlangsamt.
- Außerdem vermerken Sie bitte auf der uns zugesandten Bestellung „Public Domain".
- Zu dem Kostenbeitrag müssen wir leider pro Scheck eine Scheckgebühr von DM 0,50 berechnen. Verwenden Sie deshalb nur einen Scheck, auf dem die Gesamtsumme steht (z. B. für 2 Disketten = DM 20,50).

Auf Wunsch können Sie auch nach den alten Versandbedingungen bestellen, die wir noch zusätzlich aufrechterhalten.

Die Bezugsadresse lautet:

ST-Computer-Redaktion, Postfach 59 69, 6236 Eschborn 1

Tips und Tricks
zur ATARI Festplatte SH 204

Viele Anwender, die versucht haben, den Harddisk-Treiber der PD-Disk 20 zu installieren, werden einem Problem begegnet sein: Das Programm meldet meist, daß kein Platz mehr für den Treiber auf der Harddisk vorhanden ist. Einfachste Abhilfe: Beim Installieren der Partition setzt man den Zähler der ersten Partition (meist als Laufwerk C angemeldet) auf Null und zählt dann wieder hoch. Das hab jedoch den Nachteil, daß ca. 300 kB von der Platte verloren gehen. Eine etwas umständlichere, aber Speicherplatz sparende Möglichkeit ist es, das WINCAP-File des HDK.PRG zu ändern. Dazu wird das WINCAP-File mit einem Texteditor geladen (z. B. 1st_Word) und in der Tabelle die Stelle geändert, an der die voreingestellten Größen der jeweiligen Partition angegeben sind. Man sollte sich nun entscheiden, wie man seine Platte einteilen will. Dazu schaut man im WINCAP-File nach, an welcher Stelle die Einteilung der gewünschten Partition vorliegt und zieht dort am ersten Zahlenwert „6" (kB) ab. Nun startet man das HDX-Programm und teilt die Partitions dementsprechend ein. Nun kann das Harddisk-Programm gestartet und die Festplatte damit AUTOBOOT-FÄHIG gemacht werden. Dies hat jedoch den Nachteil, daß sich das Autoboot-Programm nicht mit einem Uhren-Accessory verträgt; der Rechner gibt öfters die Meldung aus, daß für diese Anwendung zu wenig Speicherplatz zur Verfügung steht (obwohl eigentlich noch einige 100 kB frei sind). Es kann auch passieren, daß man Laufwerke anklickt und einige Dateien nicht angezeigt werden. Dieser Effekt tritt auch bei Treibern anderer Firmen auf. Mit einem kleinen Trick läßt sich Abhilfe schaffen. **Dazu muß das AHDI-Programm**, das normalerweise von Diskette gestartet wird (durch einen Auto-Ordner auf der Boot-Disk der Festplatte), **als erstes in einen Auto-Ordner auf die Partition kopiert werden, von der gebootet werden soll.** Wird dies beachtet, dürften eigentlich keine Probleme mehr beim Betrieb mit der ATARI-Festplatte auftreten.

Als Anhang ist noch ein WINCAP-File abgedruckt, in dem bereits die Partitioneinteilung für alle voreingestellten Werte geändert ist.

(UE)

```
# Hard disk format and partition configuration file.
#  2-Apr-1986 lmd, Atari Corp.

#
# Global parameters:
#       ms#count        Maximum partition size in sectors (***DON'T CHANGE***)
#
:@@=Parameters:ms#32767:

#
# Format parameters:
# 'mn' entries (default values in [brackets]):
#       mn=name         appears in menu, name of format parms
#       hd#count        number of heads [4]
#       cy#count        number of cylinders [612]
#       lz#n            landing zone [4]
#       rt#n            seek rate code [2]
#       in#n            interleave factor [1]
#       rw#cylno        reduced-write current cylinder [cy + 1]
#       wp#cylno        write precomp cylinder [cy + 1]
#       pt=name         name of default partition to use [4-6-10]
#       sp#count        number of sectors/track [17]
#
20 Mb Mitsubuishi       :mn=SH204:
10 Mb Mitsubuishi       :mn=SH104:hd#2:pt=10:

#
# Partition selections
# 'pr' entries ('N' is '0', '1', '2', or '3'):
#       pr=name         name of partition
#       pN#size         size of partition N in bytes [0]
#       fN#number       flag byte for partition N [0 if no pN, 1 otherwise]
#       iN=ccc          id bytes for partition N [GEM]
#
# Numbers are in bytes; may be followed by 'k' (x1024) or 'm' (megabytes).
#
:pr=4-6-10:p0#4418k:p1#6m:p2#10m:
:pr=10-10:p0#10398k:p1#10404k:
:pr=16-4:p0#16377k:p1#4425k:
:pr=4-16:p1#16377k:p0#4425k:
:pr=5-5-10:p2#10562k:p0#5m:p1#5m:
:pr=5-10-5:p0#5m:p1#10562k:p2#5m:
:pr=10-5-5:p0#10562k:p1#5m:p2#5m:
:pr=5-5-5-5:p0#5440k:p1#5m:p2#5m:p3#5m:
:pr=10:p0#10m:
:pr=5-5:p0#5m:p1#5m:
```

VIP Professional im GEM-Gewand

Nach unserem ersten Test über die Tabellenkalkulation **VIP Professional** in der März-Ausgabe 1986 stellen wir nun endlich die schon damals angekündigte GEM-Version vor. Jetzt werden auch bei dieser **IBM**-identischen Software die Möglichkeiten von GEM und dem **ATARI-ST**-Rechner genutzt.

Der Lieferumfang von **VIP Professional** beinhaltet ein sehr ausführliches, deutsches Handbuch mit Anwendungsbeispielen und eine Kurzreferenz-Karte. Dazu gibt's zwei Disketten, auf denen das Haupt- und das Grafikdruckprogramm vorhanden sind. **VIP** sollte, wenn möglich, nur mit dem **TOS** in ROMs gestartet werden, da der Hauptspeicher sonst kaum für vernünftiges Arbeiten ausriecht. Man kann wahlweise auf einem monochromen oder farbigen Monitor arbeiten; auch ein Austauschen von Dateien, die in verschiedenen Auflösungen erstellt wurden, ist möglich.

Die Menü-Zeile, die nun auch von der Maus unterstützt wird, kann nach wie vor auch über die Tastatur aufgerufen werden. Die Hilfe-Funktion wird von der Maus vollständig unterstützt. Sie ist jedoch bei Anwendern, die mit Diskettenstationen arbeiten, nicht gerade schnell, da die einzelnen Hilfspunkte jedesmal neu nachgeladen werden. Hier sowie beim Laden und Abspeichern von Worksheets ist der Harddisk-Besitzer immer im Vorteil.

Das Work-Sheet

Das Arbeitsblatt hat nun die Form eines GEM-Fensters. Jetzt kann man auch Accessories, z. B. einen Taschenrechner, mitladen. Zu viele Accessories gehen freilich auf Kosten des freien ARbeitsspeichers. Das Bildschirmscrolling mit der Maus geht leider sehr langsam vor sich. So bleibt nur der Griff zur Tastatur oder die Funktion **GEHE NACH** mit der genauen Angabe der anzuspringenden Zelle.

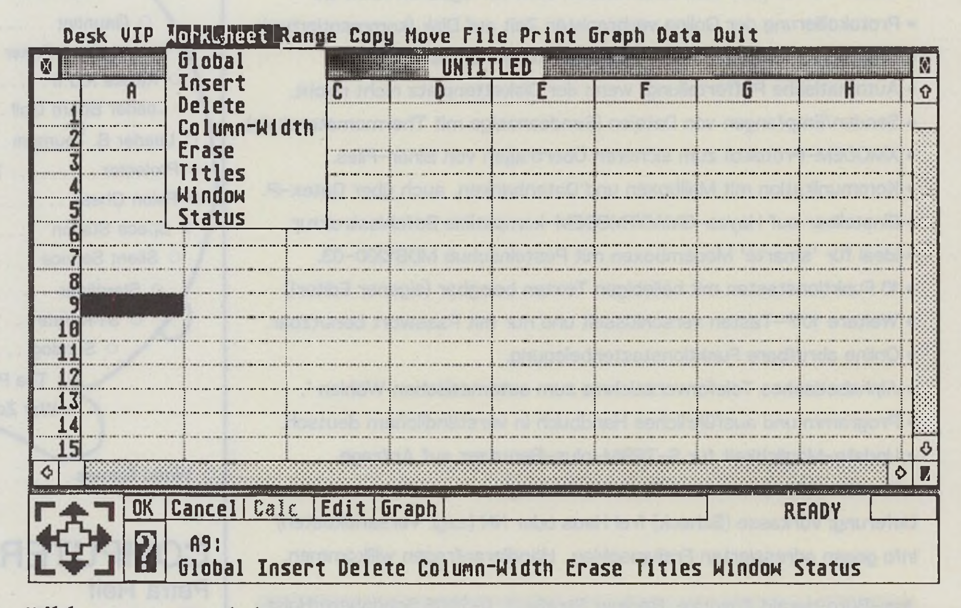


Bild 1: Das Worksheet

Eingeteilt ist das Arbeitsblatt in 8192 Zeilen ★ 256 Spalten. Davon sind ca. 16 Zeilen ★ 8 Spalten sichtbar. Den Schnittpunkt einer Zeile und Spalte bezeichnet man als Zelle. Die Zelle kann eine Breite zwischen 1 und 72 Zeichen haben, davon ist abhängig, wieviele Zellen man jeweils auf dem Bildschirm sehen kann. Es ist möglich, gleichzeitig mit dem Arbeitsblatt eine Business-Grafik als zweites Fenster darzustellen. Dabei bekommt man veränderte Werte sofort grafisch angezeigt. Doch dazu später mehr.

ATARI contra IBM (ATARI-PC)

Mit **VIP** erstellte Dateien können auf einem IBM-PC oder Kompatiblen weiterverarbeitet werden. Hierzu stellt VIP zwei Dateiformate zur Verfügung: Das ST-eigene NAT-Format, das im Laden und Speichern schneller ist, und das WKS-Format (für Lotus 1-2-3 gedacht), das aber um einiges langsamer ist. Eine Konvertierung zwischen beiden Dateitypen erledigt VIP von selbst.

Leider haben wir beim Testen ein großes Manko festgestellt: VIP ist nicht in der Lage, Unterverzeichnisse vernünftig zu verwalten. Wenn man Dateien in einem Ordner hat, kann man sich nur mit Backslash und Namen helfen. Das kommt daher, daß nicht die gewohnte Fileselektorbox erscheint und man sich deshalb den gesamten Pfadnamen merken muß. Hier sollte schnellstens Abhilfe geschaffen werden.

Mathematische Formeln und Funktionen

VIP bietet eine Vielzahl von mathematischen und finanzmathematischen Formeln und Funktionen. Sie sind hier im einzelnen aufgelistet:

Logische Funktionen

@ FALSE
Wert 0 (falsch)

@ TRUE
Wert 1 (wahr)

@ ISNA(Zahl)
1(wahr) oder 0(falsch)

ISERR(Zahl)

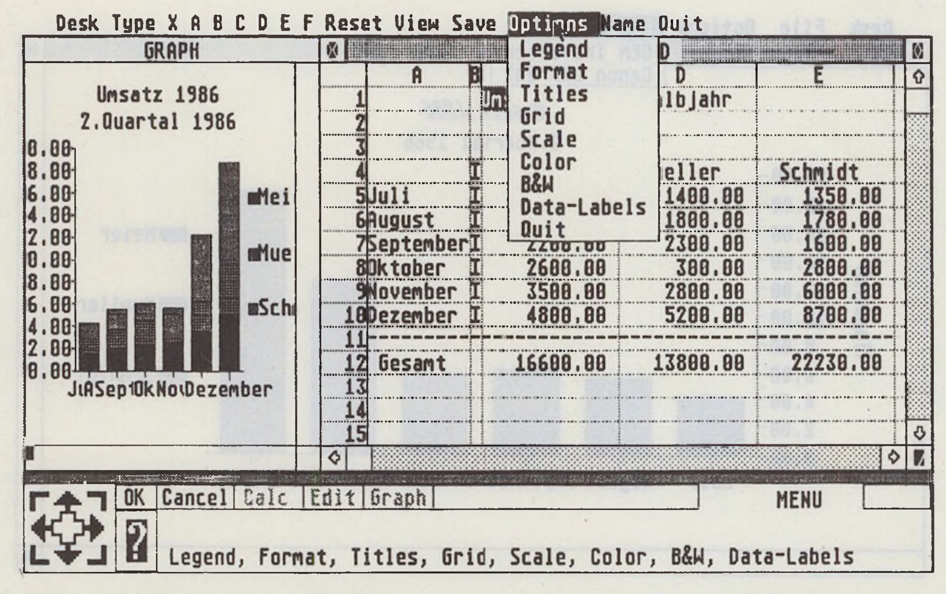


Bild 2: Worksheet und Grafik

1(wahr) wenn Wert undefinierbar oder 0(falsch)

@ IF (Zahl, Arg1, Arg2)
Wert Argument 1; wenn Zahl nicht 0; Wert Argument 2; wenn Zahl gleich 0.

Mathematische Funktionen

@ ACOS(Zahl)
Berechnet Arcus-Cosinus

@ ASIN(Zahl)
Berechnet Arcus-Sinus

@ ATAN(Zahl)
Berechnet Arcus-Tangens

@ ATAN2(Zahl1,Zahl2)
Berechnet Arcus-Tangens 4-Quadranten

@ COS(Zahl)
(Winkel in Radian) Berechnet Cosinus

@ PI
Gleich pi (3.14159)

@ SIN
(Winkel in Radian) Berechnet Sinus

@ TAN
(Winkel in Radian) Berechnet Tangens

@ ABS(Zahl)
Berechnet Absolutwert von Zahl

EXP(Zahl)
Berechnet e-funktion mit dem Parameter Zahl

@ INT(Zahl)
Ermittelt Vorkommastellen einer Zahl

@ LN(Zahl)
Berechnet natürl. Logarithmus einer Zahl

@ LOG(Zahl)
Berechnet mit Basis 10 Logarithmus einer Zahl

@ MOD(Zahl1,Zahl2)
Rest von Zahl1 durch Zahl2

@ RAND
Ermittelt eine Zufallszahl zwischen 0,0 und 1,0

@ ROUND(Zahl1,Zahl2)
Rundet Zahl1 mit 2 Dezimalstellen

@ SQRT(Zahl)
Berechnet Quadratwurzel einer Zahl

Statistische Funktionen

@ COUNT(Liste)
Anzahl der Werte in einer Liste

@ SUM(Liste)
Summe aller Zahlen in einer Liste

@ AVG(Liste)
Errechnet Mittelwert einer Liste

@ MIN(Liste)
Minimum aller Zahlen einer Liste

@ MAX(Liste)
Maximum aller Zahlen einer Liste

@ STD(Liste)
Standardabweichung aller Zahlen einer Liste

@ VAR(Liste)
Varianz aller Zahlen einer Liste

Datum-Funktionen

@ DATE
(Jahr, Monat, Tag) Berechnet Tageszahl zw. 2 Daten

@ TODAY
Verändert Tages-Datum zum fortlaufenden Datum (1–31)

@ DAY
(Tageszahl) Berechnet Kalender-Tag vom fortlaufenden DAtum (1–31)

@ MONTH
(Tageszahl) Berechnet Kalender-Monat vom fortlaufenden Datum (1–31)

@ YEAR
(Tageszahl) Berechnet Kalender-Jahr vom fortlaufenden Datum (z. B. 1987)

Spezielle Funktionen

@ NA
Nicht verfügbar

@ ERR
Wert des Fehlers

@ CHOOSE
(x,Werte) Wählt einen Wert, basierend auf einer Position einer Liste

@ HLOOKUP
(x,Zeilenbereich,Versatz) Tabellenverweis horizontal, der x mit den Werten im Zeilenbereich vergleicht.

@ VLOOKUP
(x,Spaltenbereich,Versatz) Tabellenverweis vertikal, der x mit den Werten im Spaltenbereich vergleicht.

Die Business-Grafik

Da ein Bild oft mehr aussagt als Worte oder Zahlen, ist **VIP Professional** mit einer Grafikoption ausgestattet. Sie dient dazu, Zahlen in Grafiken aufzubereiten und auf dem Monitor oder Drucker sichtbar zu machen. Natürlich ist **VIP** auch in der Lage, die Bilder auf Diskette zu speichern.

Es gibt bei VIP fünf verschiedene Typen von Grafik; maximal sind sechs Werte in einer Grafik darstellbar:

- Balken
- Gestapelte Balken
- Linien
- XY
- Kuchengrafik

Natürlich ist es möglich, die Grafiken mit Titeln und Texten zu versehen. Das Print-Out-Programm, das die Grafiken auf den Drucker überträgt, ist ebenfalls vollständig GEM-unterstützt. Leider sind seltsamerweise nur zwei Drucker implementiert, nämlich ein 9-Nadel-Matrix Drucker und ein Canon-Laser-Drucker.

An die Besitzer von 18- und 24-Nadeldruckern wurde nicht gedacht. So blieb uns beim Testen nur wieder der Griff zum Hardcopytreiber für den NEC P6. Nur so erreichten wir eine gute Qualität der Grafiken für eine Präsentation, ohne gleich einen Laserdrucker anschließen zu müssen. Natürlich ist auch das Erstellen von Overhead-Folien möglich. Auch ein Farbdrucker oder ein Plotter sollte mit in das Print-Out-Programm aufgenommen werden, um so farbige Charts auf Papier oder Folie zu bringen.

Macros (Automatischer Programmablauf)

Neben den schon aufgezählten Features bietet **VIP Professional** eine integrierte Programmiersprache. Man nennt sie **„Macros"** oder auch **„Tastaturmacros"**. Die Vorteile liegen auf der Hand: Statt jedesmal die gleichen Tastaturfolgen einzutippen, genügt eine einmalige Definition der Befehlsfolge – selbst komplizierte Funktionen werden dann auf Tastendruck ausgeführt. Auch ist es möglich, **Macrosequenzen** beim Laden des Arbeitsblattes automatisch zu starten. Alle Befehle von VIP sind auch unter den Macros lauffähig.

Integrierte Datenbank

Die Datenbank von **VIP Professional** ist eine weitere Anwendungsmöglichkeit. Mit ihr kann man nach bis zu zwei Schlüsseln gleichzeitig suchen oder sortieren lassen. Leider ist die Größe der Datenbank begrenzt, nämlich auf die Arbeitsblattgröße und die des freien Arbeitsspeichers. Eine einmal erzeugte Datenbank kann immer wieder neu bearbeitet werden. Alle Datenbank-Kommandos sind in die anderen Arbeitsblatt-Befehle integriert und können auch mit diesen kombiniert werden. Jeder Datenbanksatz besteht aus Feldern, in denen gleichartige Informationen untergebracht sind. Allerdings ist die Handhabung der Datenbank etwas umständlich; hier ist vielleicht noch eine Verbesserung angebracht.

Datenbank-Funktionen

Alle Datenbank-Funktionen werden als statistische Funktionen behandelt:

@ DFUNKTION
(Datenbankbereich, Spaltennummer, Kriterienbereich) Feldnamen immer eingeschlossen. Wenn Formeln im Bereich auftreten, müssen sie absolut sein.

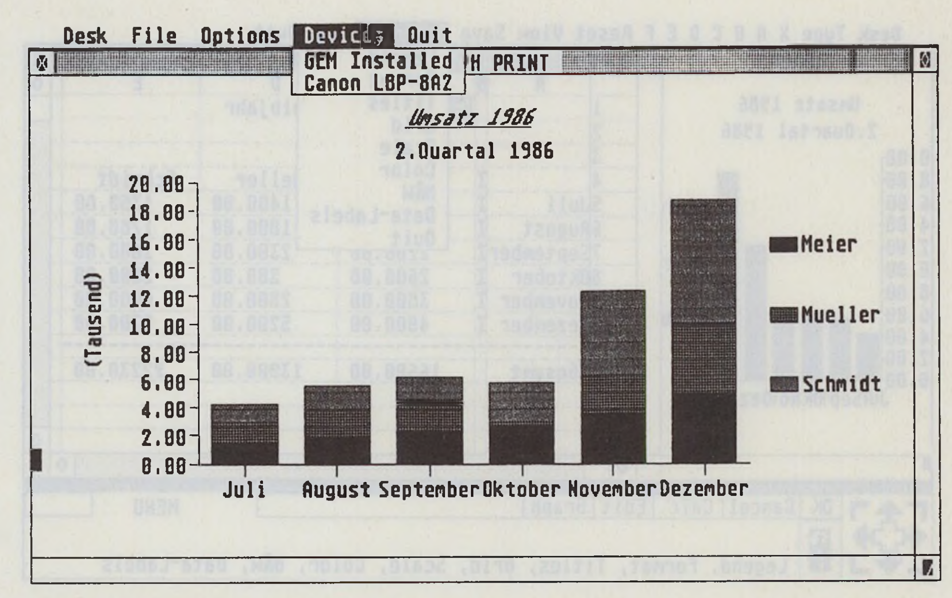


Bild 3: Das Printout Programm

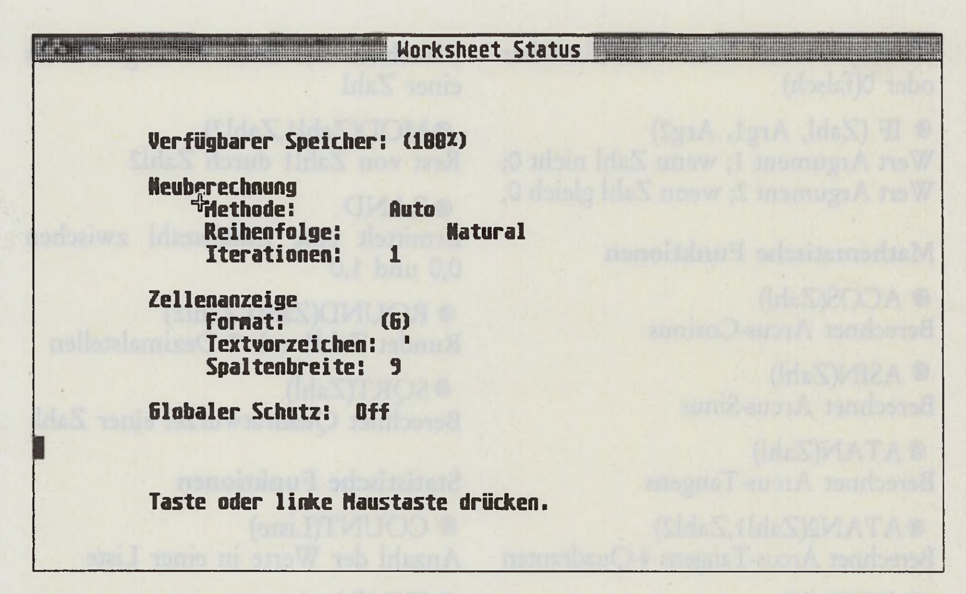


Bild 4: Statusanzeige für Worksheet

@ DCOUNT
(Datenbankbereich, Spaltennummer, Kriterienbereich) Anzahl der gewählten Werte

@ DSUM
(Datenbankbereich, Spaltennummer, Kriterienbereich) Summe des Bereiches

@ DAVG
(Datenbankbereich, Spaltennummer, Kriterienbereich) Durchschnitt des Datenbankbereiches

@ DMIN
(Datenbankbereich, Spaltennummer, Kriterienbereich) Kleinster Wert im Datenbankbereich

@ DMAX
(Datenbankbereich, Spaltennummer, Kriterienbereich) Größter Wert im Datenbankbereich

@ DSTD
(Datenbankbereich, Spaltennummer, Kriterienbereich) Standardabweichung im Datenbankbereich

@ AVR
(Datenbankbereich, Spaltennummer, Kriterienbereich) Varianz im Datenbankbereich

FAZIT

VIP Professional ist eine professionelle Tabellenkalkulation mit Datenbank- und Grafikoption. Sie ist schnell und relativ einfach zu erlernen und bietet dem Anwender ein mächtiges Werkzeug, um mit Zahlenmaterial zu operieren. In der GEM-Version ist VIP von der Bedienung her einfacher, hat jedoch noch einige Mängel. Die Pull-Down-Menüs sind zwar in Englisch, aber mit deutschen Untertiteln versehen.

Lobenswert ist die volle Kompatibilität zu LOTUS 1-2-3, so daß jeder Anwender, der beruflich schon mit LOTUS arbeitet, sofort und ohne Probleme umsteigen kann. Hier hebt sich VIP schon vom Anschaffungspreis (DM 848,–) von LOTUS 1-2-3 (ca. 1900 DM) ab – von der Hardware gar nicht erst zu reden.

(Klaus Heuer)

LOTUS 1-2-3 ist eingetragenes Warenzeichen der LOTUS DEVELOPMENT GMBH.
IBM ist eingetragenes Warenzeichen der International Business Machines.

Einkaufsführer

Hier finden Sie Ihren Atari Fachhändler

Anzeigenschluß für die April-Ausgabe ist der 27.02.87

4000 Düsseldorf

HOCO
EDV ANLAGEN GMBH
Flügelstr. 47
4000 Düsseldorf
Tel. 02 11 - 77 62 70

4050 Mönchengladbach



Hindenburgstr. 249
4050 Mönchengladbach
Tel. 0 21 61 - 187 64

4200 Oberhausen

KAMP
Büro- und Computersysteme
Vestische Straße 89/91
4200 Oberhausen 12 (Osterfeld)
Fernruf-(02 08) 89 00 86
Fernschreiber 8 56 578

4290 Bocholt

Systemlösungen
A. W.-Karlen
Alfred-Flender-Str. 284 · 4290 Bocholt
Tel. 0 28 71 - 18 34 89
OKIDATA-Fachhändler

4300 Essen

Computerservice
Keese
Haus-Berge-Straße 163 · 4300 Essen 11
(02 01) 66 70 62

ST-Computer Einkaufsführer
Werbewirksam, aktuell und preiswert.
Sprechen Sie mit uns:
Heim-Verlag
0 61 51 / 5 60 57

ATARI Systemfachhändler



KARSTADT Aktiengesellschaft
Limbecker Platz 4300 Essen 1
Tel.: (02 01) 17 63 99

4400 Münster

BASIS
COMPUTER SYSTEME GMBH
Daimlerweg 39 - 4400 Münster
Telefon 02 51 / 71 99 75 - 9

4422 Ahaus

ATARI · Epson · Fujitsu Molecular · NCR · Tandon · Schneider · Star
OCB

OCB-Computershop	OCB-Hard- und Software
Wallstraße 3	Wessumerstraße 49
4422 Ahaus	4422 Ahaus
Tel. 0 25 61/50 21	Tel. 0 25 61/50 21

4430 Steinfurt

ATARI SCHNEIDER STAR NEC SEIKOSHA PANASONIC EPSON
Computer
Büromaschinen
Service
Telefon 0 25 51 / 25 55
Tecklenburger Str. 27 · 4430 Steinfurt

4500 Osnabrück

Heinicke-Electronic
Kommenderiestr. 120 · 4500 Osnabrück
Telefon 05 41 - 8 27 99
Wir liefern Micro-Computer seit 1978

4600 Dortmund

Bürostudio BOLZ
Brauhausstraße 4 · 4600 Dortmund
Telefon 02 31 - 52 77 13 - 16



Atari, Genie, Schneider, Tandy, Brother, Star, Memorex, BASF, Verbatim
cc Computer Studio GmbH
Software-Hardware-Beratung
Service-Eilversand
Ihre Ansprechpartner:
v. Schablinski
Jan P. Schneider
Elisabethstraße 5
4600 Dortmund 1
T. 0231/528184 · Tx 822631 cccsd

City Elektronic
Güntherstraße 75
4600 Dortmund
Telefon 02 31/57 22 84

4600 Dortmund



4650 Gelsenkirchen-Horst

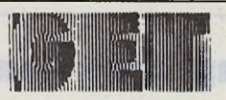


Hard- und Software, Literatur
Bauteile, Service, Versand
Groß- und Einzelhandel
Poststr. 15 · 4650 Gelsenkirchen-Horst
Tel. 0209/52572

4700 Hamm

computer center
RÜTER
Büromaschinen
Büroorganisation
Büromöbel
Werkstatt/Service
Moderne Büroausstattung
Heinrich Rüter GmbH & Co KG
4700 Hamm 1
Gustav-Heinemann-Straße 19/21
Telefon 02381/1404-0

4790 Paderborn



GESELLSCHAFT FÜR ELEKTRONISCHE TELEKOMMUNIKATION
IM SCHILDERN 15 TEL. (0 52 51) 2 60 41
4790 PADERBORN BTX ★51051#

4800 Bielefeld

hardware
software
organisation
service
CSF
CSF COMPUTER & SOFTWARE GMBH
Heeper Straße 106 – 108
4800 Bielefeld 1
Tel. (05 21) 6 16 63

Carl-Severing-Str. 190
4800 Bielefeld 14
Telefon: 05 21/45 99-150
Telex : 9 37 340 krab d
Telefax: 05 21/45 99-123
MICROTEC
Software
Hardware
Beratung
Service

4410 Warendorf

DIGITAL WORKS
Brünebrede
4410 Warendorf
Tel. 0 25 81 - 6 11 26

4930 Detmold

Frevert-Computer
Autorisierter Atari System-Händler
4930 Detmold
Bismarckstr. 12 **Tel. 05231-22416**
4920 Lemgo **Tel. 05261-88520**
Braker Mitte 9 **88440**
Autorisierter Commodore-Systemhändler
für Lippe, Minden-Lübecke u. Herford

5000 Köln

BÜRO MASCHINEN
braun
AM RUDOLFPLATZ GmbH
5000 KÖLN 1
RICHARD-WAGNER-STR. 39
RUF: 0221/219171

5010 Bergheim

Computerstudio
HÖLSCHER
EDV-Beratung · Organisation
Programmierung · Home/Personal-Computer
Software · Zubehör · Fachliteratur
Zeppelinstr. 7 · 5010 Bergheim
Telefon 02271-62096

5060 Bergisch-Gladbach

Computer Center
Buchholzstraße 1
5060 Bergisch-Gladbach
Telefon 02202-35053

5090 Leverkusen

Rolf Rocke
Computer-Fachgeschäft
Auestraße 1
5090 Leverkusen 3
Telefon 02171/2624

5200 Siegburg

Computer Center
Luisenstraße 26
5200 Siegburg
Telefon 02241/66854

5222 Morsbach

Computersysteme von A–Z
multicomp
Postanschrift: s. Hofmeister
Talstr. 22 · 5222 Morsbach-Wallerhausen
Telefon: 02294/7308

5400 Koblenz

SCHMITT COMPUTERSYSTEME
Casinostraße 40
5400 Koblenz
✆ 0261-36528

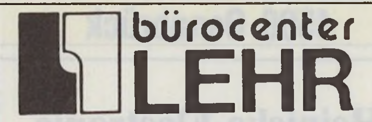
5412 Ransbach

Computer Technik Kieckbusch GmbH
Der Softwarespezialist
Am Seeufer 11+22 · 5412 Ransbach
Telefon 02623-1618

5457 Straßenhaus

DR. AUMANN GMBH
Computer-Systeme
Schulstr. 12
5457 Straßenhaus
Telefon 02634-4081/2

5500 Trier



Güterstr. 82 - 5500 Trier
☎ 0651-25044
Fordern Sie unsere Zubehör-Liste an.

5540 Prüm

ATC COMPUTER
J. ZABELL
Kalvarienbergstr. 34
5540 PRÜM
– Tel.: 06551-3483 –

5600 Wuppertal

Jung am Wall
Wall 31–33
5600 Wuppertal 1
Telefon 0202/450330

5630 Remscheid

C O M SOFT
Scheiderstr. 12 · 5630 Remscheid
Telefon (02191) 21033-34

5800 Hagen

ATARI
wir machen Spitzentechnologie preiswert.
Vertragshändler **Axel Böckem**
Computer + Textsysteme
Elper Str. 60 (Eilpezentrum) · 5800 Hagen
Tel. 02331/73490

5900 Siegen



Siegen · Weidenauer Str. 72 · ☎ 0271/73495

6000 Frankfurt

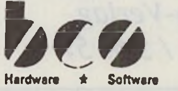
Müller & Nemecek
Kaiserstraße 44
6000 Frankfurt/M.
Tel. 069-232544

WAIZENEGGER
Büroeinrichtungen
Kaiserstraße 41
6000 Frankfurt/M.
☎ 069/7203060

5600 Wuppertal

MEGABYTE
Computer Vertriebs GmbH
Friedrich-Engels-Allee 162
5600 Wuppertal 2 (Barmen)
Telefon (0202) 81917

ATARI
... wir machen Spitzentechnologie preiswert.
Vertragshändler
Jetzt bei uns.



Büro-Computer + Organisations GmbH
Oederweg 7–9
6000 Frankfurt/M. 1
☎ (069) 550456-57
Hardware ★ Software ★ Beratung ★ Service

SCHMITT COMPUTERSYSTEME
Grosse Friedbergerstr. 30
6000 Frankfurt
✆ 069-284065

6100 Darmstadt

Heim
Büro- und Computermarkt
Heidelberger Landstraße 194
6100 Darmstadt-Eberstadt
✆ 0 61 51 / 5 60 57

ATARI Systemfachhändler



KARSTADT Aktiengesellschaft
Elisabethenstr. 15 · 6100 Darmstadt
Luisencenter · Tel. 0 61 51 - 10 94 20

SCHMITT COMPUTERSYSTEME
Mühlstraße 76
6100 Darmstadt
✆ 0 61 51 - 2 45 74

6200 Wiesbaden

SCHMITT COMPUTERSYSTEME
Rheinstraße 41
6200 Wiesbaden
✆ 0 61 21 - 30 73 30

6240 Königstein

KFC COMPUTERSYSTEME
Wiesenstraße 18
6240 Königstein
Tel. 0 61 74 - 30 33
Mail-Box 0 61 74 - 53 55

6300 Gießen



Schneider ATARI Commodore
BAUMS
BÜRO · ORGANISATION
Bahnhofstr. 26 · 6300 Gießen
Telefon: 06 41 / 7 10 96

6330 Wetzlar



Fachmarkt für Computer u. Unterhaltungs-electronic in Wetzlar,
Einkaufszentrum Bahnhofstraße, Tel. (0 64 41) 4 85 66

6400 Fulda

Schneider ATARI Commodore
WEINRICH
BÜRO · ORGANISATION
Ronsbachstraße 32 · 6400 Fulda
Telefon: 06 61 / 4 92 - 0

6457 Maintal

Landolt-Computer
Beratung · Service · Verkauf · Leasing
Wingertstr. 112
6457 Maintal/Dörnigheim
Telefon 0 61 81 - 4 52 93

6500 Mainz

:ELPHOTEC
Computer Systeme
Ihr Atari Systemhändler mit eigenem Service-Center
Schießgartenstraße 7
6500 Mainz
Telefon 0 61 31 - 23 19 47

Hier könnte
Ihre Anzeige
erscheinen.
Anruf genügt:
Heim-Verlag
✆ 0 61 51 - 5 60 57

SCHMITT COMPUTERSYSTEME
Karmeliterplatz 4
6500 Mainz
✆ 0 61 31 - 23 42 23

6520 Worms

ORION
Computersysteme GmbH
Friedrichstraße 22
6520 WORMS
Tel. 0 62 41 / 67 57 - 67 58

6700 Ludwigshafen

MKV Computermarkt
Bismarck-Zentrum
6700 Ludwigshafen
Telefon 06 21 - 52 55 96

6720 Speyer

MKV Computermarkt
Gilgenstraße 4
6720 Speyer
Telefon 0 62 32 - 7 72 16

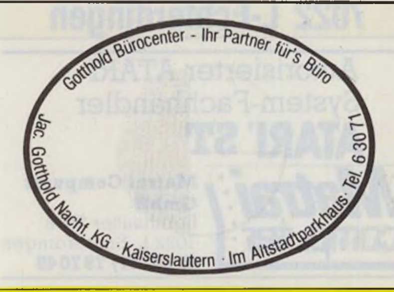
6730 Neustadt

Felten & Meier
Computersysteme
Atari + OKIDATA Fachhändler
Exterstr. 4 · 6730 Neustadt
Tel. 0 63 21 / 8 89 94

6750 Kaiserslautern

C.O.S.
COMPUTER ORG. GmbH
Karl-Marx-Straße 8
6750 Kaiserslautern
Telefon (06 31) 6 50 61 - 62

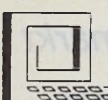
ST-COMPUTER
Sonderheft Nr. 1:
über 180 Seiten,
bei Ihrem Fachhändler



6800 Mannheim

GAUCH+STURM
Computersysteme + Textsysteme
6800 Mannheim 24
Casterfeldstraße 74-76
☎ (06 21) 85 00 40 · Teletex 6 211 912

6800 Mannheim



Computer-Center
am Hauptbahnhof GmbH

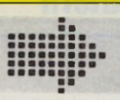
L 14, 16-17
6800 Mannheim 1
Tel. (06 21) 2 09 83/84

6900 Heidelberg

JACOM COMPUTERWELT
Hardware · Software
Schulung · Service
Mönchhofstraße 3 · 6900 Heidelberg
Telefon 0 62 21/41 05 14-550

Heidelberger Computer-Center
Bahnhofstraße 1
6900 Heidelberg
Telefon 0 62 21/2 71 32

7000 Stuttgart



BNT
COMPUTERFACHHANDEL
Der Computer-spezialist
star ATARI olivetti NEC
BNT · Computerfachhandel GmbH
Marktstraße 48 · 7000 Stuttgart 50
Mailbox 07 11/55 83 92
Datex P 45400091120
Telefon 07 11/55 83 83
Service 07 11/55 83 91
Tx 05 1933 521 dmbox g.
ref: box: dmz: bnt

SCHMITT COMPUTERSYSTEME
Tübingerstr. 18
7000 Stuttgart

7022 L-Echterdingen

Autorisierter ATARI-System-Fachhändler
ATARI ST
Matrai computer
Matrai Computer GmbH
Bernhauser Str 8
7022 L.-Echterdingen
☎ (07 11) 79 70 49

7030 Böblingen

mca Computer Center
Sindelfinger Allee 1
7030 Böblingen
Tel. 0 70 31/22 36 18

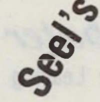
7070 Schwäb. Gmünd



Schwerzerallee 23
7070 Schwäbisch Gmünd
Telefon 07171/5633

7100 Heilbronn

Unser Wissen ist Ihr Vorteil
Walliser & Co.
Mönchseestraße 99
7100 Heilbronn
Telefon 07131/60048



Computer-Welt
Am Wollhaus 6
7100 Heilbronn
Tel. 0 71 31-6 84 01-03

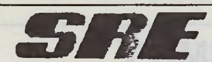
7150 Backnang



7410 Reutlingen

Computer-Shop
Werner Brock
autorisierter Fachhändler f. ATARI, Schneider, Oki
Federnseestr. 17 · 7410 Reutlingen
Telefon: 0 71 21/3 42 87

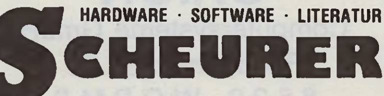
7450 Hechingen



Gesellschaft für Datenverarbeitung mbh
Computer · Drucker
Zubehör · Fachliteratur
Schloßplatz 3 · 7450 Hechingen
Telefon 0 74 71/1 45 07

7475 Meßstetten

Ihr ATARI-Systemhändler im Zollern-Alb-Kreis
HEIM + PC-COMPUTERMARKT
HARDWARE · SOFTWARE · LITERATUR



ATARI COMMODORE CUMANA DATA-BECKER
MULTITECH RITEMAN SCHNEIDER THOMSON
7475 Meßstetten 1 · Hauptstraße 10 · 0 74 31/6 12 80

7480 Sigmaringen



7500 Karlsruhe

papierhaus erhardt
Am Ludwigsplatz · 7500 Karlsruhe
Tel. 07 21-2 39 25

MKV Computermarkt
Rüppurer Straße 2d
7500 Karlsruhe
Telefon 07 21-37 30 71

7530 Pforzheim

DM Computer GmbH
Hard- & Software
Durlacherstr. 39
Tel. 0 72 31-1 39 39
7530 Pforzheim
Telex 783 248

ST-Computer Einkaufsführer
Werbewirksam, aktuell und preiswert.
Sprechen Sie mit uns:
Heim-Verlag
0 61 51/5 60 57

7600 Offenburg

FRANK LEONHARDT ELECTRONIC
Ihr Fachgeschäft für Microcomputer · Hifi · Funk
In der Jeuch 3
7600 Offenburg
Telefon 07 81/5 79 74

7640 Kehl/Rhein



Computer-Software-Marketing
eigener Service · eigene Software
Badstrasse 12
Tel. 0 78 07/8 22
Telex: 752 913
7607 NEURIED 2
Filiale:
Hauptstrasse 44
Tel. 0 78 51/18 22
7640 KEHL/RHEIN
ELEKTRO-MÜNTZER GmbH

8170 Bad Tölz

Elektronik Center Bad Tölz
Wachterstraße 3
8170 Bad Tölz
Telefon 0 80 41 / 4 15 65

8220 Traunstein

computer studio
BÜROMASCHINEN
Ludwigstraße 3 — 8220 Traunstein
Stadtplatz 10 · Tel. 0861-14767 o. 3905

8330 Eggenfelden



Computer-Centrum
R. Lanfermann

Schellenbruckstraße 6
8330 Eggenfelden
Telefon 0 87 21 / 65 73

Altöttinger Straße 2
8265 Neuötting
Telefon 0 86 71 / 7 16 10

8400 Regensburg

C-SOFT GMBH
Programmentwicklung & Hardware
Holzfällerstraße 4
8400 Regensburg
Telefon 09 41 / 8 39 86

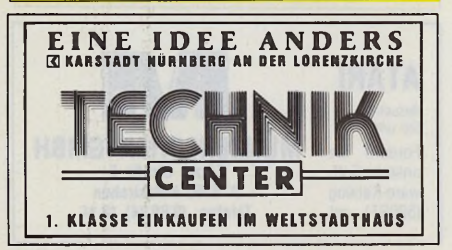
Zimmermann elektroland
8400 Regensburg — 8390 Passau
Dr.-Gessler-Str. 8 — Meraner-Str. 5
☎ 09 41 / 9 50 85 — ☎ 08 51 / 5 10 16

8490 Cham



Auf der Schanze 4 · 8490 Cham/Opf. · Telefon (09971) 9723

8500 Nürnberg



8500 Nürnberg

SCHMITT COMPUTERSYSTEME
Innere Laufer Gasse 29
8500 Nürnberg
☏ 09 11 - 20 97 17

ATARI
Der ATARI - Systemfachhändler in Nürnberg
Softwareberatung und Entwicklung
Computer • Drucker • Zubehör • Fachliteratur



HIB Computer GmbH
Äußere Bayreuther Str. 72
8500 Nürnberg 21
Tel.: 0911/515939

8520 Erlangen



Computerservice Decker
Meisenweg 29 - 8520 Erlangen
Telefon 0 91 31 / 4 20 76



8520 Erlangen
Dresdener Str. 5
Friedrichstr. 9
Tel.: 0 91 31 / 1 20 10

Alle bisher erschienenen Ausgaben ST-Computer gibt's natürlich bei Ihrem Fachhändler

8600 Bamberg



8670 Hof

COMPUTER-CENTER-BURGER
Spezialist für Personal- und Home Computer, Programme Zubehör, Beratung, Service
8670 Hof · Leimitzer Straße 11 · Telefon 09281/40075 Abt. Computer



8700 Würzburg

HALLER GMBH
Fachgeschäft für Mikrocomputer
Büttnerstraße 29
8700 Würzburg
Tel. 09 31 / 1 67 05



Hardware · Software
Service · Schulung

computer center
am Dominikanerplatz
Ruf (09 31) 5 04 88

8720 Schweinfurt

Uhlenhuth GmbH
Computer + Unterhaltungselektronik
Albrecht-Dürer-Platz 2
8720 Schweinfurt
Telefon 0 97 21 / 65 21 54

8753 Mömbris



ATARI 520 ST
APRICOT
IBM Komp
Festplatten

Hardware
Software
Schulung

Im Kahltal Zentrum, 8752 Mömbris
Telefon (0 60 29) 65 20 oder 1410

ST-Computer Einkaufsführer

Werbewirksam, aktuell und preiswert.

Sprechen Sie mit uns: Heim-Verlag 0 61 51 / 5 60 57

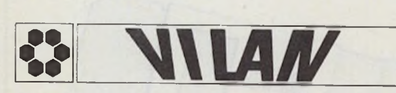
CH-8021 Zürich

Computer-Center P. Fisch

Stampfenbachplatz 4
8006 ZÜRICH
✆ 01 / 363 67 67

CH-8006 Zürich





Das Warenhaus der neuen Ideen

Ihr Computer-Fachhandel an der Bahnhofstrasse 75 · Zürich

CH-8050 Zürich



CH-8246 Langwiesen

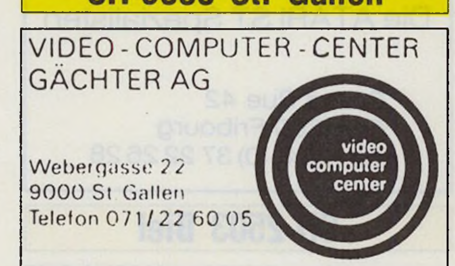
ZIMELEC

CAR-AUDIO C
COMPUTER + ELECTRONIC
Bachstraße 28 · 8200 Schaffhausen
Tel.: 053/55224

Montag—Freitag	9.00—12.00
	13.30—18.30
Samstag	9.00—16.00

CH-9000 St. Gallen

VIDEO - COMPUTER - CENTER
GÄCHTER AG



CH-9400 Rorschach

PAUS-electronic
Ihr Computer-Fachhandel
Industriestr. 30

CH-9400 Rorschach ✆ 071-41 18 85

CH-9450 Altstätten

PAUS-electronic
Ihr Computer-Fachhandel

Ringgasse 27 · CH-9450 Altstätten
Telefon 071-75 34 79

Luxemburg



J E T Z T   A U F   D I S K E T T E

Das Siegerprogramm unseres Musikwettbewerbes

Komponieren Sie
Ihr eigenes Musikstück
mit dem **MUSIK-EDITOR**

Hau rein Zottl

nur 44,– DM
zuzügl. 5,– Versandk.

unverb. empf. Verkaufspreis



DM 44,–

- Sie erhalten eine Diskette samt deutscher Anleitung.
- Auf der Diskette befinden sich neben dem Musik-Editor noch einige Musikstücke.
- Bei Bestellung verwenden Sie bitte die im Heft beigefügte Buch- u. Software-Bestellkarte.

Bestellungen bei Ihrem ATARI-Händler (s. Einkaufsführer) oder direkt beim Verlag

Heim-Verlag · Heidelberger Landstr. 194 · 6100 Darmstadt-Eberstadt · Tel. 0 61 51 / 5 60 57

Assemblerkurs

Teil 4

In diesem Teil des Assemblerkurses werde ich Ihnen die letzten Gruppen des 68000 Befehlssatzes vorstellen. Damit hätten wir die Adressierungsarten und den Befehlssatz abgeschlossen.

Zur Übersicht, gebe ich Ihnen an dieser Stelle, die Bezeichnungen der Gruppen, die in diesem letzten Teil des Kurses besprochen werden.

- Schiebe- und Rotier-Befehle
- Bitmanipulations-Befehle
- Datenübertragungs-Befehle
- Programmsteuer-Befehle

Schiebe- und Rotier-Befehle

Mit dieser Gruppe von Befehlen werden Operanden um ein oder mehrere Bits nach links oder rechts verschoben. Dies kann in einer Reihe oder im Kreis erfolgen . Die verschiedenen Befehle unterscheiden sich nur durch die Verwendung der Bits, die aus der (ea) heraus- und hereingeschoben werden.

ASL
Arithmetisches Schieben links

ASR
Arithmetisches Schieben rechts

LSL
Logisches Schieben nach links

LSR
Logisches Schieben nach rechts

ROL
Rotation nach links

ROR
Rotation nach rechts

ROXL
Rotation mit X-Bit nach links

ROXR
Rotation mit X-Bit nach rechts

Arithmetisches Schieben ASL,ASR

Beim arithmetischen Schieben des Operanden werden die Bits, die herausgeschoben werden, im C- und X-Flag gespeichert. Dem ASL-Befehl wird in die freigewordene Stelle immer eine Null nachgeschoben. Beim ASR-Befehl wird das höchstwertigste Bit kopiert, um damit die freiwerdende Stelle zu besetzen. Mit diesen Befehlen können Byte-, Wort- und Langwortdaten verarbeitet werden, nur wenn sich der Operand im Speicher befindet. Dann kann man nur mit Wortlänge arbeiten. Eine recht interessante Variante bietet der Befehl, mit einem Datenregister als Ziel. Hier ist es möglich über ein weiteres Datenregister oder einer Konstante, die Anzahl der Verschiebungen vorzugeben.

Beispiel:

```
                        C-Flag
D1  =  %10011101          X
           ASL.B #4,D1
D1  =  %11010000          1
           ASR.B #4,D1
D1  =  %11111001          1
```

Wendet man beide Beispiele auf das Register D1 an, so erscheint, nach Ausführung der Befehle, das entsprechende Ergebnis (siehe D1 letzte Zeile). Die Konstante kann Werte zwischen eins und acht annehmen. Ein Schieben um eine Stelle nach links, entspricht einer Multiplikation mit zwei. (nach rechts, dementsprechend durch zwei). Das Vorzeichen der Zahl bleibt bei diesen Operationen erhalten. Findet durch das Schieben ein Vorzeichenwechsel statt, so wird dies in dem V-Flag vermerkt.

Logisches Schieben LSL,LSR

Bei diesen zwei Befehlen gibt es nur einen winzigen Unterschied, im Vergleich zu dem ASR-Befehl. Der ASL-Befehl ist identisch mit dem LSL-Befehl. Während beim ASR-Befehl das Vorzeichenbit dupliziert wird, wird beim LSR-Befehl einfach eine null nachgeschoben.

Rotieren ROL,ROR

Das Bit, daß jeweils aus dem Operanden herausgeschoben wird, wird im C-Flag gespeichert und gleichzeitig an die freigewordene Stelle kopiert. Somit ist der Kreis gschlossen. Ebenfalls werden bei dieser Gruppe von Befehlen die Flags entsprechend gesetzt, wobei diese beiden Rotierbefehle das X-Flag nicht verändern. Als kleine Aufgabe: Wieviele Einsen sind in einem Datenregister enthalten?

Lösung:

```
        MOVE.L  #31,D3
        CLR.L   D2
LOOP    ROR.L   #1,D1
        BCS     L1
        ADDQ    #1,D2
L1      DBRA    D3,LOOP
```

Der Schleifenzähler (D3) wird mit dem Wert 31 geladen, da der DBcc-Befehl bei −1 abbricht. Das Register D2 wird gelöscht, um anschließend die Anzahl der vorkommenden Einsen zu speichern. D1 selbst enthält die zu untersuchende Zahl.

Rotieren mit dem X-Flag ROXL,ROXR

Bei diesen Befehlen erfolgt das Rotieren über das X-Flag. Das X-Flag fungiert hier praktisch als ein Zwischenspeicher. Dabei wird das Bit, daß rausgeschoben wird, ins X-Flag geschoben. Dessen Inhalt wiederum, geht an den Anfang der Rotation, um die Kette zu schließen.

- bleibt unverändert
★ wird entsprechend gesetzt
U Undefiniert
0 wird auf Null gesetzt

Bei der Schreibweise der erlaubten Adressierungsarten bedeutet:

ARI alle Adressregister indirekt
abs Absolut kurz und lang
PCR alle Programmcounter relativ
SR Statusregister
CCR Condition Code Register
USP User Stack Pointer

Um alle Adressierungsarten zu ermitteln, kann man jede Quelle mit jedem Ziel verknüpfen.

Für den Platzhalter „d“ kann hier „L oder R“ eingesetzt werden. Um die Wirkung der Befehle besser klarzumachen, habe ich noch eine grafische Darstellung gewählt, an der Sie die Verhältnisse genau ersehen können.

Bitmanipulation-Befehle

Wie der Name schon sagt, kann man mit diesen Befehlen einzelne Bits manipulieren. Deshalb haben die Befehle auch keine Operandengröße. In welcher Art und Weise man die Bits manipulieren kann, sagt Ihnen die folgende Tabelle:

- BCHG ändern
- BCLR löschen
- BSET setzen
- BTST testen

Steht der Zieloperand im Speicher, so kann nur ein Bit innerhalb eines Bytes angesprochen werden. Nur wenn ein Datenregister das Ziel der Operation ist, sind alle 32 Bit ansprechbar. Das niederwertigste Bit hat dabei die Nummer null.

Als erstes wird das spezifizierte Bit getestet. Ist es null, so wird das Z-Flag (Zero), das einzigste Bit im CCR, daß von dieser Gruppe verändert wird, auf eins gesetzt. Dies entspricht dem BTST-Befehl. Dieser Befehl wird häufig mit einem anschließenden Bcc-Befehl gebraucht. Durch diese Kombination kann man einfache Entscheidungen aufgrund eines Bits fällen. Die anderen drei Befehle bauen auf dem BTST-Befehl auf. Nachdem das entsprechende Bit getestet worden ist, wird das angesprochene Bit anschließend gesetzt (BSET), gelöscht (BCLR) oder geändert (BCHG). In der Tabelle der Adressierungsarten steht der BTST-Befehl stellvertretend für die anderen Befehle dieser Gruppe.

Datenübertragungs-Befehle

In dem zweiten Teil hatten Sie ja schon einen Teil dieser Gruppe kennengelernt. Dies war im Prinzip nur der leistungsstarke MOVE-Befehl. Die restlichen Befehle dieser Gruppe sind:

EXG
Austausch von Registerinhalten

LEA
Lade effektive Adresse

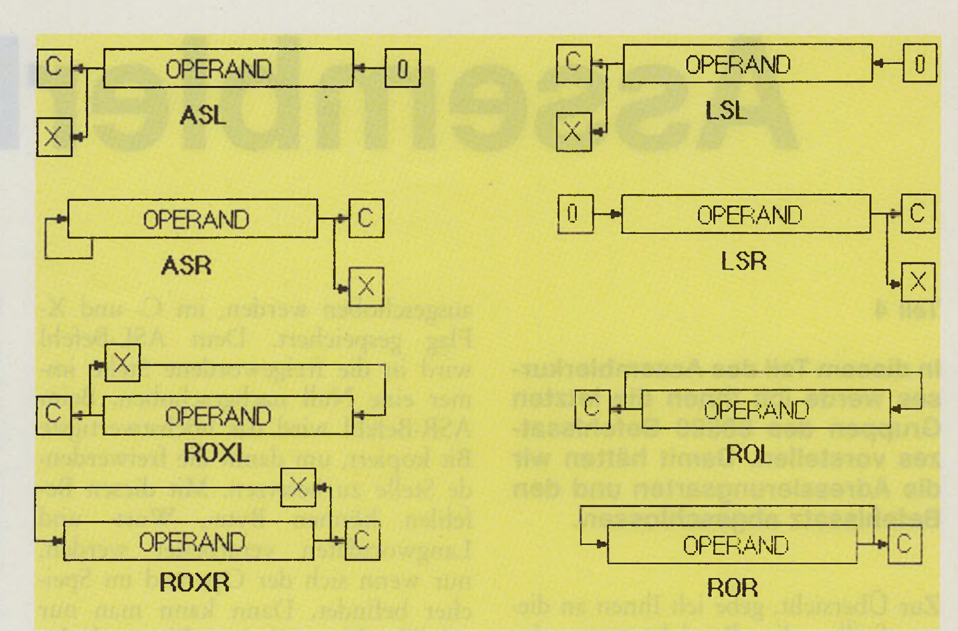


Bild 1: Schiebeoperationen

Syntax	Flags XNZVC	.x	Quelle	Ziel
ASd.x Dx,Dy	*****	B,W,L	Dn	Dn
ASd.x #Kons,Dn	*****	B,W,L	#	Dn
ASd.x (ea)	*****	W		abs,ARI
LSd.x Dx,Dy	*****	B,W,L	Dn	Dn
LSd.x #Kons,Dn	*****	B,W,L	#	Dn
LSd.x (ea)	*****	W		abs,ARI
ROd.x Dx,Dy	-**0*	B,W,L	Dn	Dn
ROd.x #Kons,Dn	-**0*	B,W,L	#	Dn
ROd.x (ea)	-**0*	W		abs,ARI
ROXd.x Dx,Dy	***0*	B,W,L	Dn	Dn
ROXd.x #Kons,Dn	***0*	B,W,L	#	Dn
ROXd.x (ea)	***0*	W		abs,ARI
BTST Dn,(ea)	--*--		Dn	Dn,ARI,abs
BTST #Konst,(ea)	--*--		#	Dn,ARI,abs

LINK
Baue Stackbereich auf

PEA
Lege Adresse auf Stack

SWAP
Vertausche Registerhälften

UNLK
Baue Stackbereich ab

Tausche Registerinhalte EXG

Manchmal ist es notendig, die Inhalte von zwei Registern auszutauschen. Dazu dient dieser Befehl. Dieses Problem wird Ihnen in einer höheren Programmiersprache schon mal begegnet sein. Zur Lösung benötigten Sie eine weitere Variable. In Assembler existiert ein Befehl, der ohne ein weiteres Register zwei Registerinhalte austauscht. Der Befehl arbeitet nur mit Registern und mit einer Operandenlänge von 32 Bit. Manche Assembler, lassen deshalb die Schreibweise EXG.L zu, obwohl dies nicht notwendig wäre.

Beispiel:

EXG D1,A2

	vorher	nachher
D1	18273645	77661254
A2	77661254	18273645

Tausche Registerhälften SWAP

Im Gegensatz zum Austausch von Registerinhalten wird hier der Inhalt eines Datenregisters vertauscht. Dies geschieht, indem die Bits 16–31 nach 0–15, und die Bits 0–15 nach 16–31 kopiert werden. Die Operation erfolgt also nur in Wortlänge. Die Flags N und Z werden nach dem 32 Bit-Ergebnis gesetzt. Das heißt, daß Z gleich eins wird, wenn die Bits im Datenregister (Bits 0–31) null sind. Entspre-

chend geht demzufolge das Bit 31 ins N-Flag.

Beispiel:
SWAP D2

	vorher	nachher
D2	1112222	2221111

Lade effektive Adresse LEA

Mit diesem Befehl kann man sich die Arbeit etwas erleichtern und die Programme übersichtlicher gestalten. LEA berechnet eine Adresse und gibt sie an ein Adressregister weiter. Somit ist die Operandenlänge auf 32-Bit festgelegt. Die Berechnung der effektiven Adresse erfolgt genauso wie bei der Adressrechnung der einzelnen Adressierungsarten. Nehmen wir einmal an, wir wollten ein Tabellenelement im Speicher bearbeiten. Die Adresse des Tabellenelements wird durch die Adressierung 7(A1,D3.L) beschrieben. Ein Zugriff auf dieses Tabellenelement sähe dann so aus.

```
MOVE.L 7(A1,D3.L),D0
```

Benötigt man mehrere Zugriffe auf diese Speicherzelle, und will man mit dieser Adressierung noch weitere Tabellenteile damit verarbeiten, so wird dies recht lange dauern. Denn jedesmal muß die Adressrechnung durchgeführt werden. Besser ist es dann schon, eine Adressrechnung auszuführen und diese Adresse in einem Adressregister zu behalten. Z. B.:

```
LEA 7(A1,D3.L),A0
MOVE.L (A0),D0
      .(A0)
```

Vergleicht man die zwei Darstellungen, die das gleiche bewirken, so merkt man, daß LEA und MOVE dieselbe effektive Adresse berechnen. LEA bringt das Ergebnis dieser Rechnung nach A0 und der MOVE den Inhalt der Adresse nach D0.

Lege Adresse auf Stack PEA

Der PEA-Befehl ist recht einfach erklärt. Er berechnet genauso wie der LEA-Befehl eine effektive Adresse, nur gibt er diese nicht an ein Adressregister weiter, sondern legt diese auf den Stack (−(A7)). Man könnte PEA entsprechend so beschreiben:

```
LEA (ea),-(A7)
```

Da dieser Befehl aber nicht existiert, könnte man ihn durch folgende Sequenz ersetzen:

```
LEA (ea),A3
MOVE.L A3,-(A7)
```

Baue Stackbereich auf LINK

Der Aufbau eines Stackbereiches erfolgt in drei Schritten.

1.) ein Adressregister auf den Stack bringen.

2.) das Adressregister mit Stackpointer laden.

3.) eine Adressdistanz auf Stackpointer addieren.

Die Zahl, die auf den Stackpointer addiert wird, ist eine 16-Bit-Zahl im Zweierkomplement, die Vorzeichenrichtig auf 32-Bit erweitert wird. Damit hat man zwei Möglichkeiten einen neuen Stackbereich anzulegen. Entweder mit einer positiven oder negativen Zahl. Meist wird eine negative Zahl benutzt, um Platz für Daten zu schaffen. Die Länge dieses Bereiches entspricht der Adressdistanz. Über diesen Datenbereich können Datenblöcke, variabel im Stack untergebracht, zwischen Haupt- und Unterprogramm ausgetauscht werden. Die Funktion soll folgende Grafik erläutern. Ein Schema zur Nutzung dieses Befehls erfolgt in diesem kleinen Beispiel:

```
; Hauptprogramm

JSR UNTERPROGRAMM
        .

; Unterprogramm
LINK A6,-$80
MOVEM.L D0-D7/A3,-(A7)
        .

        .
MOVEM.L (A7)+,D0-D7/A3
UNLK A6
RTS
```

Da das Adressregister die Adresse des alten Stack enthält, sollte man es auf dem neuen Stack abspeichern. Geht dieser Wert verloren, so kann der alte Wert dieses Registers nicht wiederhergestellt werden (was ja nicht unbedingt tragisch ist), aber der RTS-Befehl findet nicht mehr die richtige Rücksprungadresse ins Hauptprogramm!!! Somit ist das Programm rettungslos verloren.

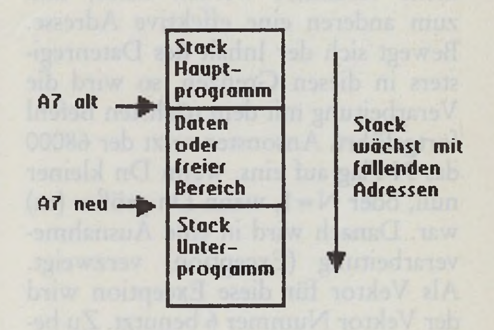
Baue Stackbereich ab UNLK

Hier haben wir das Gegenstück zu dem vorhergehenden Befehl. Was damit passieren kann, haben Sie ja schon erfahren. Warum das so ist, werden Sie mit der Funktion dieses Befehls verstehen. Der Abbau des Stackbereichs erfolgt in zwei Schritten.

1.) lade Stackpointer mit alter Adresse.

2.) lade Adressregister mit Wert vom Stack.

Sie sehen, der Schritt zurück geht nur richtig, wenn der Stackpointer mit seinem alten Wert geladen wird.



Damit hätten wir die vorletzte Gruppe der Befehle vervollständigt. Nun folgt noch die Tabelle dieser Gruppe:

Die Programmsteuerbefehle

Mit der Vervollständigung dieser Gruppe wird die Besprechung des 60000'er Befehlssatzes abgeschlossen sein. Einen Teil, Sie erinnern sich, hatte ich ja schon besprochen. Zu den besprochenen Befehlen gehörten z. B.

Syntax		Flags XNZVC	.x	Quelle	Ziel
EXG	(ea),(ea)	-----		Dn,An	Dn,An
LEA	(ea),An	-----		ARI/(An)+/-(An)	An
LEA	(ea),An	-----		abs,PCR	An
LINK	An,Kons			An	
PEA	(ea)	-----		ARI/(An)+/-(An)	
PEA	(ea)	-----		abs,PCR	
SWAP	Dn	-**00			Dn
UNLK	An			An	

die Verzweigungsbefehle. Nun wieder eine Übersicht über die Befehle.

CHK
Prüfe gegen Grenzen
NOP
Keine Operation
RESET
Rücksetzen der Peripherie
Scc
Setze nach Bedingung
STOP
Halte die Verarbeitung an
TAS
Prüfe und setze ein Bit
TRAPV
Exception mit Bedingung

Prüfen gegen Grenzen CHK

Mit diesem Befehl kann man den Inhalt eines Datenregisters, bzw. die Bits 0–15, gegen Grenzen prüfen. Die Grenzen sind zum einen die Null, die nicht verändert werden kann, und zum anderen eine effektive Adresse. Bewegt sich der Inhalt des Datenregisters in diesen Grenzen, so wird die Verarbeitung mit dem nächsten Befehl fortgeführt. Ansonsten setzt der 68000 das N-Flag auf eins, wenn Dn kleiner null, oder N=1, wenn Dn größer (ea) war. Danach wird in eine Ausnahmeverarbeitung (Exception) verzweigt. Als Vektor für diese Exception wird der Vektor Nummer 6 benutzt. Zu beachten wäre, daß nach Ausführung des Befehls alle Flags, außer dem X-Flag, undefiniert sind.

Meist findet der Befehl seinen Einsatz in Compilern. Mit ihm kann sehr einfach geprüft werden, ob der Zugriff auf eine indizierte Variable oder ein Matrizenelement zulässig ist.

Keine Operation NOP

Dies ist wohl der einfachste Befehl des 68000, denn hier arbeitet er nicht! Garnicht wäre etwas übertrieben, sonst wäre der Befehl ja nicht implementiert worden. Daran kann man sehen, das Nichtstun auch seine Berechtigung hat. So zum Beispiel kann er dazu benutzt werden, Zeitschleifen abzugleichen. Denn Rechenzeit kostet der Befehl schon. Da er auch Speicherplatz benötigt, kann man mit ihm einen Speicherbereich im Programm belegen, um diesen Teil später mit richtigen Befehlen vom Programm aus zu beschreiben.

Läßt man ein Unterprogramm mit einem NOP-Befehl anfangen, so kann sich das Unterprogramm gegen einen zweiten Aufruf schützen, wenn es den NOP-Befehl einfach mit dem Code für RTS überschreibt. Jeder weitere Aufruf führt dann zur sofortigen Beendigung des Unterprogrammes. Ebenso ist es denkbar, daß ein weiteres Programm dieses Unterprogramm wieder freigibt. Damit hat man sich eine aufwendige Verwaltung erspart.

Rücksetzen der Peripherie RESET

Da dies ein priviligierter Befehl ist, kann er nur im Supervisormodus ausgeführt werden. Er ermöglicht es, die an die Resetleitung des 68000 angeschlossene Peripherie zurückzusetzen. Dies macht er, indem er die Leitung für 124 Clockzyklen auf „Low“ (ca. null Volt) legt. Danach befinden sich alle angeschlossenen Bausteine im Zustand nach dem Einschalten.

Setze nach Bedingung Scc

Dieser Befehl setzt ein Byte (nur Byteverarbeitung) in Abhängigkeit einer Bedingung (cc). Die Bedingungen sind die gleichen wie bei dem DBcc-Befehl. Ist die Bedingung erfüllt, so wird das über die (ea) adressierte Byte mit Einsen gefüllt, also auf $FF gesetzt. Ansonsten wird es auf $00 gesetzt.

Beispiel:

SEQ D3

	vorher	nachher
Z=1 (Bed. erfüllt)		
D3	XX	00
Z=0 (Bed. nicht erfüllt)		
D3	XX	FF

Halte die Verarbeitung an STOP

Dies ist ebenfalls ein priviligierter Befehl. Wird er nicht im Supervisor-Modus ausgeführt, so geht der 68000 in eine Exception. Der Ausdruck hinter dem STOP-Befehl ist eine 16 Bit Zahl. Diese Zahl wird benutzt, um das Statusregister damit zu laden. Ist dies geschehen, so geht der 68000 in den Halt-Zustand. Der Prozessor nimmt seine Arbeit wieder auf, wenn:

- ein Interrupt mit einer höheren Priorität als die Interruptmaske gewählt wird.
- ein Reset ausgelöst wird, da dieser die höchste Priorität hat.
- das Tracebit eins war.

Tritt ein sogenannter katastrophaler Fehler auf, der nicht behoben werden kann, so setzt man den Prozessor auf Halt. Über die Interruptmaske, können laufende Prozesse, die sich an den Prozessor wenden, gesperrt werden. In den Flags könnte man eine Nummer ablegen, die später Auskunft über den Stoppunkt gibt. Somit kann man dann auf den Fehler im System schließen.

Prüfe und setze ein Bit TAS

Als erstes wird der 8-Bit lange Operand getestet, und danach die Flags entsprechend gesetzt. Dann wird vom Operand das 7. Bit gesetzt, egal ob dieses null oder eins war.

Beispiel:

TAS $1000

	vorher	nachher
1000	00	80
1000	87	87

Exception mit Bedingung TRAPV

Den TRAP-Befehl hatte ich ja schon besprochen. An den TRAPV-Befehl wird allerdings nur eine kleine Bedingung geknüpft, bevor dieser in die Exceptionbehandlung geht. Der Befehl fragt erst das V-Flag ab. Ist dieses gesetzt, so wird TRAP ausgeführt.

Da wir jetzt die 56 Befehle des 68000 Befehlssatzes besprochen haben, gebe ich Ihnen noch eine Tabelle der priviligierten Befehle. Um diese Befehle nutzen zu können, müssen Sie zuerst die TOS-Funktion SUPER aufrufen. Diese bringt Sie in den Supervisormodus.

- ANDI to SR
- EORI to SR
- MOVE to SR
- MOVE USP
- ORI to SR
- RESET
- RTE
- STOP

Es gibt allerdings noch einen „Befehl“ den der 68000 verarbeitet. Allgemein wird er als ILLEGAL bezeichnet. Trifft der 68000 auf eine Bitfolge, die er nicht als Befehl interpretieren kann, so löst er eine Exception aus. Diese Exception hat die Vektornummer 4. Manche Assembler haben diesen „Be-

fehl" als ILLEGAL implementiert, obwohl Motorola keinen Assemblersyntax dafür vorgesehen hat.

Das Programm

Das Programm, daß ich Ihnen zum Abschluß dieses Assemblerkurses vorstellen möchte, soll Ihnen den Rahmen für die GEM-Programmierung in Assembler geben. In diesem Rahmen befindet sich ein recht bekanntes Programmbeispiel. Eine Alertbox! Im Prinzip geht es nicht um diese Alertbox, sondern um die Verwaltung der AES und VDI Variablen. Da wir uns auf der untersten Ebene der Programmiersprachen bewegen, ist die Verwaltung in Assembler etwas aufwendiger als zum Beispiel in C.

Die erste kleine Routine hat den Namen Setblock. Dies ist ein TOS Aufruf, wie Sie ihn schon kennen. Diese Routine gibt den nicht benötigten Speicherplatz an das Betriebssystem zurück. Dazu wird aus der Base Page (256 Bytes), die vom Betriebssystem zur Verfügung gestellt wird, die Länge des Programms berechnet. Außerdem wird der Userstackpointer auf das obere Ende des neuen Bereichs gesetzt. Den alten Stackpointer wird der Setblock-Routine zur Verwahrung mitgegeben. Diese Prozedur ist nötig, wenn ein weiteres Programm oder ein RSC-File in den Speicher geladen werden soll. Denn das Betriebssystem verwaltet ja den Speicher.

Mit einem kleinen JSR verzweigen wir in unser Programm. Dieses Programm wird zur ordnungsgemäßen Terminierung mit einem RTS abgeschlossen. Allerdings habe ich auch ein gewaltsames Verlassen des Programms eingebaut, nämlich mit einem Sprung nach END.

Nehmen wir einmal an das Programm wird ordnungsgemäß verlassen, so wird das Programm nach dem Unterprogrammaufruf fortgesetzt. An dieser Stelle findet man den TOS Aufruf zur Eingabe eines Zeichens. Dann folgt die Marke END, mit der anschließenden Funktion TERM, die das Programm beendet.

Da ich im Rahmen dieses Assemblerkurses nicht auf die GEM-Programmierung eingehen kann, werde ich mich hier auf die Parameterübergabe beschränken. Um diesen Ausführungen folgen zu können, sollten Sie sich schon ein wenig mit der GEM-Programmierung auskennen. Die Variablennamen halten sich weitgehend an die üblichen Bezeichnung.

Syntax	Flags XNZVC	.x	Quelle	Ziel
CHK (ea),Dn	-*UUU		Alle/An	Dn
NOP	-----			
RESET	-----			
Scc (ea)	-----			Alle/PCR/abs(lang)
STOP #Kons	*****		# (16 Bit)	
TAS (ea)	-**00			Alle/An/PCR
TRAPV #Kons	-----		# (0-15)	

TEIL4.PRG



AES

Wenn Sie sich einmal das kleine Unterprogramm AES anschauen, so sehen Sie, daß der AES Aufruf ein Softwareinterrupt ist. Zu diesem Zwecke werden zwei Register initialisiert. Das Register D1 enthält eine Adresse und D0 die Geheimnummer von AES. Die Sache sieht bis jetzt noch ganz einfach aus. Wenn Sie das Programm weiter verfolgen, und zu der Stelle AESPB (am Anfang vom Datenbereich) kommen, stehen die Adressen von weiteren Marken. Dies sind die Adressen der Felder zur Übergabe der Variablen-Felder an AES. Diese Felder muß der Benutzer entsprechend seinen Wünschen mit Daten versorgen. Wenn Sie richtig gezählt haben, so macht dies sechs Felder. Jedes Feld hat eine bestimmte Bedeutung.

Das CONTRL Feld

Dieses Feld enthält alle Informationen über die Art der Funktion, sowie die Größe der Eingabefelder. Es besteht aus fünf Worten (10 Bytes). Hier erfolgt auch die Rückmeldung über die Größe der Ausgabefelder. Das Feld ist folgendermaßen aufgebaut:

contrl
Befehlsnummer

sintin
Größe des intin-Feldes in Bytes

sintout
Größe des intout-Feldes in Bytes

saddrin
Anzahl der Adressen im addrin-Feld

saddrout
Anzahl der Adressen im addrout-Feld

Die Einträge contrl, sintin und saddrin müssen von Ihnen gemacht werden. Rückmeldungen über die Anzahl der Einträge in den Feldern, die an Sie gehen, geben die Variablen sintout und saddrout an.

Das GLOBAL Feld

Dieses Feld enthält einige Daten über die Applikaton. Mit diesem Feld werden Sie wahrscheinlich nie in Berührung kommen.

Die anderen Felder müßten Ihnen bekannt vorkommen. Basic beispielsweise, stellt Ihnen diese Variablen zur Verfügung. Wenn Sie nun noch beachten, daß diese Felder Wortlänge haben, so kann nicht mehr viel schiefgehen. Definiert wurden die Felder mit 128 Worten. Wollen Sie größere Datenmengen auf einmal verarbeiten, so vergrößern Sie diese.

VDI

Wenn Sie sich den Zeiger VDIPB anschauen, und das Unterprogramm VDI, so sehen Sie nicht viel Neues. Der Zeiger enthält diesmal nur 5 Einträge. Das CONTRL-Feld wird hier ähnlich benutzt. Da VDI mehr Einträge im CONTRL-Feld benötigt (Insgesamt 12 Wörter), habe ich saddrout mit acht Wörter reserviert. Die Funktion des CONTRL-Feldes sieht nun folgendermaßen aus:

contrl
Befehlsnummer

contrl+2
Anzahl der Koordinatenpaare im ptsin-Feld

contrl+4
Anzahl der Koordinatenpaare im ptsout-Feld

contrl+6
Anzahl der Wörter im intin-Feld

contrl+8
Anzahl der Wörter im intout-Feld

contrl+10
Unterfunktion-Befehlsnummer

contrl+12
Anwendungskennziffer

Die Variablen contrl, contrl+2, 6, 10 und 2 werden dem VDI übergeben. Als Rückmeldung enthalten contrl+4, 8 und evtl. contrl+12 einen Wert.

Die Felder intin und intout werden von beiden Funktionen gemeinsam benutzt.

Da dies die Verwaltung der Variablen war, können wir uns nun mit dem Hauptprogramm befassen. Das Hauptprogramm beginnt mit vier Aufrufen. Zwei davon sind AES. Diese Funktionen dienen zur Initialisierung der Applikation. Als erstes, wird die Applikation mit Applikation Init angemeldet. Danach benutzt man die Funktion Graf Handle, um von ihr die Variable grhandle zu erhalten. Diese Variable benötigen Sie für die VDI-

```
; Dieses Programm soll Ihnen die Anwendung der VDI und AES
; Routinen zeigen. Das Programm beinhaltet hauptsächlich
; die Verwaltung der Routinen. Diesen Teil sollten Sie sich
; auf Diskette abspeichern, um sich unnötige Tipparbeit bei
; weiteren Programmen zu sparen. Aus Schönheitsgründen habe
; ich die Funktion CLEAR WORKSTATION nicht benutzt.
;
 move.l   a7,a5             ; Stackpointer speichern
 move.l   #nstack,a7        ; neuen Stack setzen
 move.l   4(a5),a5          ;
 move.l   $c(a5),d0         ; Textsegment
 add.l    $14(a5),d0        ; Datensegment
 add.l    $1c(a5),d0        ; Blocksegment
 add.l    #$100,d0          ; base page
 move.l   d0,-(a7)          ; Speicherplatzbedarf
 move.l   a5,-(a7)          ; alter Stackpointer
 move     #0,-(a7)          ; dummy
 move     #$4a,-(a7)        ; TOS  SETBLOCK
 trap     #1                ; Aufruf
 adda     #12,a7            ; Stackkorrektur
;
 jsr   main                 ; Hautprogramm
;
 move     #1,-(a7)
 trap     #1                ; wartet auf Taste
 addq.l   #2,a7
end
 clr.l    (a7)
 trap     #1                ; beendet Programm
;
aes                         ; AES Aufruf
 move.l #aespb,d1           ; Pointer
 move   #$c8,d0             ; AES Nummer
 trap   #2                  ; doit
 rts
;
vdi                         ; VDI Aufruf
 move.l #vdipb,d1           ; Pointer
 move.l #$73,d0             ; VDI Nummer
 trap   #2                  ; doit
 rts
;
main                        ; Anfang der Initialisierung
 clr.l   ap1resv
 clr.l   ap2resv
 clr.l   ap3resv
 clr.l   ap4resv
 move    #10,opcode
 clr     sintin
 move    #1,sintout
 clr     saddrin
 clr     saddrout
 jsr     aes                ; Applikation Init
;
 move    #77,opcode
 clr     sintin
 move    #5,sintout
 clr     saddrin
 clr     saddrout
 jsr     aes                ; Graf Handle
;
 move    intout,grhandle    ; grhandle von Routine speichern
;
 move    #100,opcode
 clr     contrl+2
 move    #11,contrl+6
 move    grhandle,contrl+12
;
 lea      intin,a1          ; Intin (0-9) löschen
 move    #9,d1
loop
 move    #1,(a1)+
 dbra d1,loop
;
 move    #2,(a1)
 jsr     vdi                ; Open virtuell Workstation
;
 jmp nloesch                ; Clear Workstation überspringen
;
 move    #3,contrl
```

```
 clr     contrl+2
 clr     contrl+6
 move    grhandle,contrl+12
 move    #1,intin
 jsr     vdi                        ; Clear Workstation
;
nloesch
;
; Hier kann nun ihr Programm folgen
;
 move.l  #alert,addrin              ; Alarmtext nach addrin
box
 move     #52,contrl
 move     #1,contrl+2
 move     #1,contrl+4
 move     #1,contrl+6
 move     #0,contrl+8
 move     #1,intin                  ; 1. Knopf ist Returnknopf
 jsr      aes                       ; Form Alert
;
 move    intout,d0                  ; Rückeldung der Taste
 cmp     #1,d0
 beq end                            ; Taste 1, dann ende
 cmp     #2,d0
 bne weiter                         ; taste 2, weiter. mit Returntaste beenden
 move.l #alert2,addrin              ; 2. Alarmtext nach addrin
 jmp     box                        ; und ausführen
weiter
;
 rts                                ; ende von main
;
 .data
 .even
; Daten zu den Alertboxen
alert
; Pictogramm (0-3)
 dc.b "[1]"
; Text 5 Zeile a 40 Zeichen durch | getrennt
 dc.b "[Dies ist eine|Alertbox in Assembler]"
; Bis zu drei Tasten. Pro Taste maximal 20 Zeichen
 dc.b "[Taste 1|Taste 2|Taste 3]",0,0
 .even                              ; Wichtg, da ungerade Adresse wahrscheinlich
;
alert2                              ; dito
 dc.b "[2]"+
 dc.b "[Sie haben die Taste 2|gedrückt]"
 dc.b "[ende]",0,0
 .even
;
; Hier endet Ihr Programm
;
; Daten zu der Initialisierung
;
aespb                               ; Pointer auf AES Felder
 dc.l contrl,global,intin,intout,addrin,addrout
;
vdipb                               ; Pointer auf VDI Felder
 dc.l contrl,intin,ptsin,intout,ptsout
;
grhandle       ds.w 1               ; Speicher Graf Handle
;                        FELDER
contrl
opcode         ds.w 1
sintin         ds.w 1
sintout        ds.w 1
saddrin        ds.w 1
saddrout       ds.w 8               ; noch 7 Worte für AES
;
global
apversion      ds.w 1
apcount        ds.w 1
apid           ds.w 1
apprivate      ds.l 1
aptree         ds.l 1
ap1resv        ds.l 1
ap2resv        ds.l 1
ap3resv        ds.l 1
ap4resv        ds.l 1
;
intin          ds.w 128
;
ptsin          ds.w 128
;
intout         ds.w 128
;
ptsout         ds.w 128
;
addrin         ds.w 128
;
addrout        ds.w 128
;
 .bss
 .even
 ds.l 300                           ; 1200 Bytes müssten reichen
nstack                              ; Neuer Stackpointer
```

Aufrufe.

Da wir den Bildschirm benutzen möchten, müssen wir beim ATARI dies durch ein Open Virtuell Workstation dem Betriebssystem mitteilen. Als Dank erhalten Sie dann insgesamt 57 Werte. Diese Werte enthalten z. B. Daten über das Auflösungsvermögen des Monitors, Linien- und Schrifttypen usw..

Und als letzte Funktion Clear Workstation, die ich aus Schönheitsgründen nicht benutzt habe, löscht einfach den Bildschirm.

Jetzt sind wir endlich mit der Initialisierung fertig, und können unseren Ideen freien Lauf lassen. Die Alertbox, ein beliebtes Beispiel, wird deshalb so gern benutzt, weil GEM die komplette Verwaltung dafür übernimmt. Dazu gehört das Zwischenspeichern des benötigten Bildschirmbereiches, die Aufbereitung des Aussehens der Box, nebst ihrer Größe, bis hin zur Restaurierung des Bildschirms. Als Antwort bekommt man eine Zahl in intout zurück, die die Nummer der gedrückten Taste enthält.

Diese Information bringe ich zur weiteren Auswertung ins Register D0. Wurde die Taste 1 betätigt, so endet das Programm mit einem gewaltsamen Abbruch. Taste 3 hingegen beendet das Programm normal. Hier kann man schön die Restaurierung des Bildschirmes erkennen, da erst noch eine weitere Taste gedrückt werden muß, bis sich das Desktop wieder aufbaut. Die Taste 2 quitiere ich mit einer neuen Alertbox. Diese Box hat als einzigsten Ausgang nur das brutale Ende zur Folge.

Ich hoffe, ich konnte Ihnen mit diesem Assemblerkurs die Assemblersprache ein wenig näher bringen. Wenn Sie sich in Assembler ein wenig eingearbeitet haben, so dürftes es Ihnen keine Probleme bereiten Beispiele oder Anwendungen, die in Assembler programmiert wurden, zu verstehen und für sich selbst zu nutzen. Sind Sie mit der Materie etwas besser vertraut, so sollte Ihnen die Anpassung solcher Programme zu Ihren eigenen Zwecken unproblematisch sein.

Sven Schuler

ISAM & PRIMA

(Teil 4)
– Die ISAM-Routinen

In der letzten Folge unserer ISAM & PRIMA-Serie kommen wir nun zu dem Teil, der sich mit der Index-sequentiellen Datenzugriffsmethode (ISAM) beschäftigt. Zunächst sei noch einmal kurz erklärt, was eine Index-sequentielle Datei ist.

Eine ISAM-Datei kann wahlfrei (also mit Direktzugriff auf einen ganz bestimmten Datensatz) oder auch sequentiell (vorwärts und rückwärts) verarbeitet werden, ausgehend sowohl von Dateianfang bzw. -ende als auch von jeder beliebigen Stelle innerhalb der Datei. Selbstverständlich können Sätze gelesen, geändert und an dieselbe Stelle der Datei zurückgeschrieben werden.

Ferner lassen sich Datensätze einfügen, wodurch sich die Datei an beliebiger Stelle ergänzen läßt. Löscht man einen Datensatz, so muß dieser im Datenteil als gelöscht gekennzeichnet und entsprechend behandelt werden. Doppelte Schlüsselwörter (Keys) sind in unserem ISAM nicht vorgesehen. D. h., man kann nur noch einem Auswahlkriterium auf einmal suchen und keine Schlüssel miteinander verknüpfen.

Als Index wird lediglich ein sechs Bytes langes Feld am Anfang jedes Datensatzes benötigt. Auf dieses Indexfeld muß unmittelbar das Schlüsselwort folgen, an das sich die übrigen Satzdaten anschließen können.

Der Geschwindigkeitsvorteil, der durch die Verwendung eines indexsequentiellen Datenzugriffs entsteht, ist enorm und wird bei Verwendung einer Hard- oder RAM-Disk noch gesteigert.

Unser ISAM beinhaltet nur zwei Macros, wodurch die Anwendung nicht allzu schwer fallen sollte. Es ist so eingerichtet, daß maximal drei verschiedene Dateien (1–3) auf einmal eröff-

```
        KEYLEN:     MACRO
                    MOVE.L      ?2,ISAM?1+8
                    MOVEM.L     D1-D3,-(SP)
                    MOVE.L      EOF?1,D1
                    MOVE.L      RECL?1,D3
                    JSR         DIVI
                    MOVE.L      D1,ISAM?1+4     ;LOF NACH QX
                    MOVEM.L     (SP)+,D1-D3
                    ENDM
```

KEYLEN

```
        ISAM:       MACRO
                    MOVEM.L     A0-A2,-(SP)
                    MOVE.L      #ISAM?1,A1
                    MOVE.L      ?3,A2
                    LEA         QROUTVEC+?2,A0
                    MOVE.L      (A0),A0
                    JSR         (A0)
                    MOVEM.L     (SP)+,A0-A2
                    ENDM
                    DC          "*",0
                    POS         #5,#11
                    CONTROL     CZEILEND
                    PRINT       "with ISAM Expansion"
                    POS         #0,#15
                    DELAY       #5000,#800
                    JMP         ISAMEND
```

ISAM

```
; * DEFINITIONEN FÜR ISAM
;
;
QI:         EQU     0           ;DISPLACEMENTS IM FILE-KONTROLLBLOC
QX:         EQU     4
QKEYLEN:    EQU     8
QIDENT:     EQU     12
;
QV:         EQU     0           ;DISPLACEMENTS IM USER-BUFFER
QL:         EQU     2
QR:         EQU     4
QY:         EQU     6
;
SCHREIBEN:EQU       0           ;ISAM-FUNKTIONSNAMEN
SUCHEN:     EQU     4
LESEN:      EQU     8
GLEICH_HOCH:EQU     12
LESEN_R:    EQU     16
GLEICH_TIEF:EQU     20
;
            EVEN                ;ADRESSEN FUNKTIONSROUTINEN
QROUTVEC:   DC.L    QWRITE      ;NEUEN KEY EINFÜGEN
            DC.L    QSEARCH     ;SUCHEN BESTIMMTEN KEY
            DC.L    QREAD       ;VORWÄRTSLESEN SEQUENTIELL
            DC.L    QEQ_HI      ;GLEICHEN ODER HÖHEREN KEY SUCHEN
            DC.L    QREAD_R     ;RÜCKWÄRTSLESEN SEQUENTIELL
            DC.L    QEQ_LO      ;GLEICHEN ODER KLEINEREN KEY SUCHEN
;
QA:         BLK     101         ;AUFTRAGSKEY
QZ:         BLK     101         ;ARBEITSBEREICH
            EVEN
QIND:       DC.L    0           ;ARBEITSINDEX
QM:         DC      0           ;SCHALTER DATEIANFANG
QF:         DC      0           ;FEHLERANZEIGER
```

Die **Neuen** und die **Erfolgreichen aus dem** Heim-Verlag



Wenn Sie das Software-Paket **VIP-Professional** kaufen wollen oder schon besitzen, dann weiht Sie dieses Buch schnell und umfassend in die Geheimnisse dieses Profi-programmes ein.

VIP-Professional besteht aus den drei Funktionsbereichen
- DATENBANK
- KALKULATION
- GRAFIK

mit denen wichtige und vielfältige Aufgaben hervorragend gelöst werden.

Das Buch enthält komplette Musterlösungen für die Gewinn- u. Verlustrechnung und Fakturierung. Anhand dieser Beispiele wird gezeigt, wie Sie alles herausholen, was in VIP-Professional steckt.

Mit diesem Buch können Sie **VIP-Professional** richtig einsetzen und seine Möglichkeiten voll ausschöpfen.

B-408 Buch 49,– DM
D-428 Diskette 49,– DM



Anhand von über 130 Programmbeispielen und Erläuterungen lernen und trainieren Sie das Programmieren in BASIC. Von einfachen, aber grundlegenden Beisp. bis zur ausgereiften Anwendung findet der ATARI ST-Besitzer Beispiel-Programme, die den perf. Einstieg in die Programmiersprache des GfA-BASIC's leicht machen. Alle Beisp. werden so dargestellt, daß das Verständnis für die Programmstruktur gefördert wird, aber auch die Details der GfA-BASIC-Version deutlich werden. Die Progr. sind gut dokumentiert, wobei die hervorr. Möglichkeiten des GfA-Basic zur strukturierten Programmierung genutzt werden. **Aus dem Inhalt:** ● Eigenschaften des GfA-BASIC ● Dateiverwalt. u. Ordner in GfA-BASIC ● Schleifentechnik ● Felder ● Unterprogramme ● Menü-Steuerung ● Window Technik ● Zufallszahlen ● Seq.- u. RANDOM-Dateien ● Textverarbeitung ● Sortierprogramme ● Fakturiersysteme ● Grafik ● Spiel uvm.
Zum Buch: Programmdiskette mit sämtl. Beispielprogr.
320 Seiten mit 131 Programmen in GfA-BASIC

ATARI ST — Prof. Dr. B. Bollow / K. Reimann — **GfA-BASIC Programm-Sammlung** — 131 Programme in GfA-BASIC

B-407 Buch 49,– DM
D-427 Diskette 39,– DM

Das Buch für den richtigen Einstieg mit dem ATARI ST. Leicht verständlich wird der Lernende in den Lernstoff eingeführt.

Einige der Themen:

Die Hardware des ATARI ST
- Überblick über die Systemkomponenten und ihre Funktionsweise
- Aufstellung und Wartung des Computers

Die Software des ATARI ST
- Arbeiten mit dem GEM (das Desktop/Maus/Icons etc.)
- Kopieren von Files und Disketten, Löschen und Formatieren u.v.a.m.
- Die Programmiersprachen BASIC und LOGO
- Einführung in die Bedienung und Programmierung
- Programmsammlung mit vielen interessanten Beispielen

350 Seiten mit 55 Abbildungen.

B-400 Buch 49,– DM
D-420 Diskette 39,– DM



B-402 Buch 49,– DM
D-422 Diskette 39,– DM

Der ideale Einstieg in die Programmierung mit LOGO. Ein Buch für alle, die LOGO schnell verstehen und perfekt erlernen möchten. Ein Buch wie es sein muß, leicht zu verstehen und interessant geschrieben.

Einige der Themen:
- **Variablen- und Listenverarbeitung**
- **Packages und Properties**
- **Die Turtlegrafik**
- **Viele ausführlich erklärte Beispielprogramme zu den Themen**
  - Arbeiten mit relativen und sequentiellen Dateien
  - Anwendungen aus Mathematik
  - Hobby- und Spielprogramme u.v.a.m.

Zum Buch gibt es die Programmdiskette mit sämtlichen Beispielprogrammen.

ca. 350 Seiten mit vielen Abbildungen.

Ein Grundwerk für alle ATARI ST-Besitzer. Auf über 300 Seiten eine klare und verständliche Einführung in die Programmiersprache BASIC, elementare BASIC-Kommandos, Diskettenhandhabung und vieles, was zur perfekten Beherrschung des ATARI ST gehört.

Ein Spitzenbuch mit über 80 Übungs- und Anwenderprogrammen wie z. B.:

Sortierprogramme / Textverarbeitung / Umgang mit sequentiellen- u. Random-Dateien / Fakturierprogramm / Programmiertechniken an ausgewählten Beispielen u. v. a. m.

Zum Buch gibt es die Programmdiskette mit sämtlichen Beispiel-Programmen.



B-401 Buch 49,– DM
D-421 Diskette 39,– DM



B-406 Buch 49,– DM
D-426 Diskette 39,– DM

C ist die zweite „Muttersprache" des Atari ST: schnell, komfortabel, kompakt im Code.

„C auf dem Atari ST" ist für Anwender geeignet, die Erfahrungen mit anderen Programmiersprachen gemacht haben. Das Buch behandelt den vollen Sprachumfang von Standard-C und verweist auf BASIC und Pascal.

Besonderer Wert wird auf die Anschaulichkeit und Genauigkeit der Darstellung gelegt. Alle Programmbeispiele sind getestet und direkt in den Text übernommen.

Aus dem Inhalt: C-Compiler für den Atari: Digital Research, Lattice, Megamax ● der Editor ● Bedienung des Compilers ● Grundlegende Elemente eines C-Programms ● Variablentypen ● Felder und Vektoren ● Ausdrücke ● Zeiger ● Speicherklassen ● Bitfelder ● Varianten ● Aufzählungen ● Dateien ● Diskettenhandling ● Einbindung von Assemblerprogrammen ● Bildschirmgrafik in C ● Fehler in den C-Compilern ● Tools u. a.

Ein Grundwerk.
Wer sich mit der Programmierung der GEM-Funktionen vertraut machen will braucht dieses Buch!
Es beginnt mit einer Erläuterung des GEM-Aufbaus und führt anhand von Programmbeispielen zum **leichten Verständnis aller auf dem Atari verfügbaren GEM-Funktionen.** Die übersichtliche Gliederung ermöglicht auch ein schnelles Nachschlagen der Funktionen.

Einige der Themen:
● Was ist **GEM** ● Die GEM-Bestandteile **VDI** und **AES** ● Die **GEM-Implementation** auf dem Atari ST ● Aufruf der GEM-Funktionen aus **BASIC, C,** und **ASSEMBLER** ● Sonstige **Programmiersprachen** und GEM ● Die Programmierung der **VDI**-Funktionen ● Die **AES**-Bibliothek und die Programmierung ihrer Funktionen ● Aufbau eines **Objektbaumes** ● Was ist eine **Resource**-Datei? ● Viele erläuterte **Beispielprogramme** in BASIC, C und ASSEMBLER.



B-404 Buch 49,– DM
D-424 Diskette 39,– DM



B-405 Buch 49,– DM
D-425 Diskette 39,– DM

Zum neuen Basic Interpreter, ein Buch, das mit gezielten Beispielen verständlich den Einstieg in das Basic der Superlative ermöglicht. Ein Muß für jeden Besitzer dieses Interpreters. Einige der Themen:
- Der Umgang mit dem Editor
- Ausführliche, mit Beisp. versehene Befehlsübersicht
- Die fantastischen Grafikmöglichkeiten (Windows, Sprites, Alertbox, Pull-down Menüs
- Strukturierte Programmierung – auch in Basic möglich!
- Dateiverwaltung unter GfA-Basic
- GEM Handhabung in Basic
- Vergleich mit anderen Basic-Interpretern – Hilft bei der Kaufentscheidung
- Zahlreiche Übungs- und Anwenderbeispiele
- Mathematik und hohe Genauigkeit

Zum Buch gibt es die Programmdiskette mit sämtlichen Beispielprogrammen.

ACHTUNG: Zu den Büchern gibt es die Programmdiskette mit allen Programmbeispielen – Diskette laden und los geht's...



6100 Darmstadt-Eberstadt
Heidelberger Landstr. 194
Telefon 0 61 51 / 5 60 57

▶ **Für Bestellungen verwenden Sie am besten die in der ST-Zeitschrift vorhandene BUCH- UND SOFTWARE-BESTELLKARTE**
▶ **Bücher und Programm-Disketten aus dem Heim-Verlag erhalten Sie auch bei Ihrem ATARI-Fachhändler oder im Buchhandel**

*alle Preise sind unverbindliche Preisempfehlungen

net werden können. Vor dem Versuch, eine weitere Datei ohne vorherige Änderung des ISAM-Sourcecodes anzusprechen, kann man nur warenen. Wem jedoch drei gleichzeitig offene ISAM-Dateien nicht genügen , dem kann durch einfaches Einfügen eines jeweils 13 Byte langen Kontrollblocks (siehe Definitionen für ISAM) im Sourcecode geholfen werden. Außerdem ist an zwei Stellen jeweils eine kleine Änderung im Programm erforderlich.

Beispiel zur Erweiterung auf eine vierte ISAM-DAtei

1. QWRT020 muß folgendermaßen aussehen:

```
QWRT020:   CMP.W   #3,QIDENT(A1)
           BNE     QWRT030
           PUT     3,QIND,A2
           RTS
QWRT030    PUT     4,QIND,A2
           RTS
```

2. QLES020 muß folgendermaßen aussehen:

```
QLES020:   CMP.W   #3,QIDENT(A1)
           BNE     QLES030
           GET     3,QIND,A2
           RTS
QLES030:   GET     4,QIND,A2
           RTS
```

Diese Änderungen müßten leicht auszuführen sein, so daß man entsprechend auf einfache Art und Weise auch eine fünfte oder sechste ISAM-

```
          EVEN
ISAM1:    BLK       12          ;KONTROLLBLOCK FILE 1
          DC.W      1           ;BLOCK-IDENTITY
ISAM2:    BLK       12          ;KONTROLLBLOCK FILE 2
          DC.W      2           ;BLOCK-IDENTITY
ISAM3:    BLK       12          ;KONTROLLBLOCK FILE 3
          DC.W      3           ;BLOCK-IDENTITY
```

```
; * ISAM FUNKTIONSROUTINEN
;
;
QWRITE:   MOVEM.L   D0-D2,-(SP)
          JSR       QKEYERW
          MOVE.L    #1,QIND
          MOVE.L    #1,QI(A1)
          CLR.B     QF
          TST.L     QX(A1)
          BNE       QW010
          MOVE.L    A2,D1
          ADD.L     #QY,D1
          MOVE.L    QKEYLEN(A1),D0
          TRANSFER  #QA,D1,D0
          MOVE.L    #1,QX(A1)
          CLR.W     QV(A2)
          CLR.W     QL(A2)
          CLR.W     QR(A2)
QWEXIT:   MOVEM.L   (SP)+,D0-D2
          RTS
QW010:    JSR       QINDLES
          MOVE.L    #QA,D1
          MOVE.L    A2,D2
          ADD.L     #QY,D2
          MOVE.L    QKEYLEN(A1),D0
          COMPARE   D1,D2,D0
          BEQ       QW030
          BHI       QW040               ;QY > QA
          TST.W     QR(A2)
          BEQ       QW020
          MOVE.W    QR(A2),D0
          EXT.L     D0
          MOVE.L    D0,QI(A1)
          MOVE.L    D0,QIND
          BRA       QW010
QW020:    ADDQ.L    #1,QX(A1)
          MOVE.L    QX(A1),D0
          MOVE.W    D0,QR(A2)
QW025:    JSR       QINDWRT
```

```
         MOVE.L     QX(A1),QIND
         MOVE.L     A2,D1
         ADD.L      #QY,D1
         MOVE.L     QKEYLEN(A1),D0
         TRANSFER   #QA,D1,D0
         MOVE.W     QI+2(A1),QV(A2)
         MOVE.L     QX(A1),QI(A1)
         CLR.W      QL(A2)
         CLR.W      QR(A2)
         BRA        QWEXIT
QW030:   MOVE.B     #1,QF
         BRA        QWEXIT
QW040:   TST.W      QL(A2)
         BEQ        QW050
         MOVE.W     QL(A2),D0
         EXT.L      D0
         MOVE.L     D0,QI(A1)
         MOVE.L     D0,QIND
         BRA        QW010
QW050:   ADDQ.L     #1,QX(A1)
         MOVE.W     QX+2(A1),QL(A2)
         BRA        QW025
QINDWRT: CMP.W      #1,QIDENT(A1)
         BNE        QWRT010
         PUT        1,QIND,A2
         RTS
QWRT010: CMP.W      #2,QIDENT(A1)
         BNE        QWRT020
         PUT        2,QIND,A2
         RTS
QWRT020: PUT        3,QIND,A2
         RTS
QKEYERW: MOVEM.L    D0/A3,-(SP)        ;QA AUF KEYLEN ERWEITERN
         MOVE.L     #QA,A3
         MOVE.L     QKEYLEN(A1),D0
         SUBQ.L     #1,D0
QKEY010: TST.B      (A3)+
         DBEQ       D0,QKEY010
         TST.W      D0
         BMI        QKEY020            ;LANGE=KEYLEN
         SUBQ.L     #1,A3              ;FÜLLADR -> A3
         MOVE.L     #QA,D0             ;ADR. QA
         ADD.L      QKEYLEN(A1),D0     ;+ KEYLEN
         SUB.L      A3,D0              ;- FÜLLADR = FÜLL-LANGE
         FILL       A3,D0,#" "
QKEY020: MOVEM.L    (SP)+,D0/A3
         RTS
QINDLES: CMP.W      #1,QIDENT(A1)
         BNE        QLES010
         GET        1,QIND,A2
         RTS
QLES010: CMP.W      #2,QIDENT(A1)
         BNE        QLES020
         GET        2,QIND,A2
         RTS
```

Datei ermöglichen kann.

Man sollte möglichst keine ISAM-Datei mit sortierten Eingabedaten erstellen, da hier die Regel „Je mehr durcheinander, desto schneller“ gilt.

In den ISAM-Macros wird die eigentliche Arbeit nicht durch die Macros KEYLEN und ISAM selbst geleistet, sondern durch die ISAM-Funktionsroutinen, mit denen der Anwender selbst gar nicht direkt zu tun hat. Diese Routinen entsprechen den Funktionen Schreiben, Suchen, Lesen, Lesen_R, Gleich_hoch und Gleich_tief.

KEYLEN
– Schlüssellänge festlegen

Format: KEYLEN Datenkanalnummer, #Länge

Die Länge des Schlüssels muß für jede ISAM-Datei nach dem OPEN- und vor dem ersten ISAM-Macro-Aufruf mittels KEYLEN definiert werden. Wenn man Schlüssel mit einer Länge von mehr als 100 Bytes verwenden will, muß man die Felder QA und QZ im Sourcelisting auf die gewünschte Schlüssellänge +1 verändern (z. B. gewünschte Länge = 110 Bytes, dann müssen QA und QZ den Wert 111 zugewiesen bekommen).

ISAM
– ISAM-Funktionsroutinen

Format: ISAM Datenkanalnummer, Funktion, #Puffer

Der Puffer muß dabei an der Wortgrenze beginnen!

Funktionen:

SCHREIBEN
(Satz einfügen)

SUCHEN
(Bestimmten Schlüssel suchen)

LESEN
(logisch nächsthöheren Datensatz lesen)

GLEICH__HOCH
(gleichen oder nächsthöheren Datensatz lesen)

LESEN__R
(logisch nächstniedrigeren Datensatz lesen)

GLEICH__TIEF
(gleichen oder nächstniedrigeren Datensatz lesen)

Was zu beachten ist!

Vor dem ISAM-Aufruf muß der Schlüssel im Feld QA stehen und mit X'00' abgeschlossen sein. Dieses X'00' hat den Vorteil, daß ein Schlüssel, dessen Länge kleiner ist als die definierte Schlüssellänge, nicht rechtsbündig mit Leerstellen aufgefüllt werden muß. ISAM erledigt das von selbst.

Ist der Aufruf erfolgt und ein Fehler aufgetreten, erhält man einen Fehlercode in QF (.B) zurück. Die Fehlercodes haben folgende Bedeutung:

1 = Datensatz schon vorhanden
2 = Datensatz nicht gefunden
3 = Ende der Datei (EOF) erreicht

Hat man bei der Funktion „Lesen" ein EOF erhalten, sollte man nicht gleich wieder versuchen, mit der Funktion Lesen__R den letzten Datensatz zu erreichen. Die bessere Methode ist, QM auf 1 zu setzen und dann erst Lesen__R aufzurufen. Das bringt dann den letzten Datensatz der Datei oder, falls gar kein Datensatz existiert, ein EOF. Folglich muß QF unbedingt abgefragt werden, da ansonsten ja z. B. nicht das Ende der Datei erkannt werden kann.

Ferner erhält man die aktuelle Datensatz (Record)-Nummer in ISAMx (.L) zurück (das x steht für die Datenkanalnummer). Der Datensatz kann vor dem Schreiben im Puffer aufbereitet und anschließend geschrieben werden. Bei allen anderen Funktionen steht der Datensatz im Puffer und kann weiter bearbeitet werden. Der Wert in ISAMx

```
QLES020:  GET        3,QIND,A2
          RTS

QSEARCH:  MOVEM.L    D0-D2,-(SP)
          JSR        QKEYERW
          CLR.B      QF
          MOVE.L     #1,QI(A1)
          MOVE.L     #1,QIND
          MOVE.L     QI(A1),D1
          CMP.L      QX(A1),D1
          BHI        QS020              ;QI > QX
QS000:    JSR        QINDLES
          MOVE.L     #QA,D1
          MOVE.L     A2,D2
          ADD.L      #QY,D2
          MOVE.L     QKEYLEN(A1),D0
          COMPARE    D1,D2,D0
          BNE        QS010
QSEXIT:   MOVEM.L    (SP)+,D0-D2
          RTS
QS010:    BHI        QS040              ;QY > QA
          TST.W      QR(A2)
          BNE        QS030
QS020:    MOVE.B     #2,QF
          BRA        QSEXIT
QS030:    MOVE.W     QR(A2),D0
          EXT.L      D0
          MOVE.L     D0,QI(A1)
          MOVE.L     D0,QIND
          BRA        QS000
QS040:    TST.W      QL(A2)
          BEQ        QS020
          MOVE.W     QL(A2),D0
          EXT.L      D0
          MOVE.L     D0,QI(A1)
          MOVE.L     D0,QIND
          BRA        QS000

QREAD:    MOVEM.L    D0-D2,-(SP)
          TST.L      QX(A1)
          BNE        QR010
          MOVE.B     #9,QF
QREXIT:   MOVEM.L    (SP)+,D0-D2
          RTS
QR010:    CLR.B      QF
          TST.B      QM
          BEQ        QR030
          CLR.B      QM
          MOVE.L     #1,QI(A1)
QR020:    MOVE.L     QI(A1),QIND
          JSR        QINDLES
          TST.W      QL(A2)
          BEQ        QREXIT
          MOVE.W     QL(A2),D0
          EXT.L      D0
          MOVE.L     D0,QI(A1)
          BRA        QR020
QR030:    MOVE.L     QI(A1),QIND
          JSR        QINDLES
          TST.W      QR(A2)
          BEQ        QR040
          MOVE.W     QR(A2),D0
          EXT.L      D0
          MOVE.L     D0,QI(A1)
          BRA        QR020
QR040:    TST.W      QV(A2)
          BNE        QR050
          CLR.L      QI(A1)
          MOVE.B     #9,QF
          BRA        QREXIT
QR050:    MOVE.L     QKEYLEN(A1),D1
          MOVE.L     A2,D2
          ADD.L      #QY,D2
          TRANSFER   D2,#QZ,D1
          MOVE.W     QV(A2),D0
          EXT.L      D0
          MOVE.L     D0,QI(A1)
          MOVE.L     D0,QIND
          JSR        QINDLES
          MOVE.L     QKEYLEN(A1),D0
          MOVE.L     A2,D1
          ADD.L      #QY,D1
          MOVE.L     #QZ,D2
          COMPARE    D1,D2,D0
          BHI        QR040              ;QZ > QY
          BRA        QREXIT
```

kann verändert werden. Auf diese einfache Art und Weise kann man einen beliebigen Datensatz zum aktuellen machen. Die Gesamtanzahl der Datensätze kann man aus ISAMx+4 (.L) abfragen. Diesen Wert darf der Benutzer auf gar keinen Fall verändern.

Abschließend noch eine Anmerkung zum Schreiben von Datensätzen: Hat man die Funktion Schreiben gewählt, muß man den entsprechenden Datensatz auch tatsächlich mit 'PUT Dateinummer,ISAMx,#Puffer' schreiben, es sei denn, QF meldet eine Eins, also 'Datensatz schon vorhanden'. Das Schlüsselfeld (Keyfeld) des Puffers hat ISAM schon selbständig aufbereitet. Dieses Feld darf nur gelesen, aber nie beschrieben werden. Dies ist ISAM selbst vorbehalten.

Mit diesem letzten Teil von ISAM & PRIMA ist unsere Serie über die Assembler-Macros beendet. Wir hoffen, daß Sie die Macros auch dann verwenden können, wenn Sie keinen SEKA-Assembler von KUMA haben. Natürlich müssen sie dann entsprechend angepaßt werden. Schlimmstenfalls muß man sie jedesmal einbinden, wenn man einen Assembler benutzt, der keine Macroverarbeitung erlaubt. Für alle Tippfaulen: Die Macros sind komplett für DM 49,– beim Heim-Verlag zu beziehen.

(HE)

```
QREAD_R:  MOVEM.L   D0-D2,-(SP)
          TST.L     QX(A1)
          BNE       QRV010
          MOVE.B    #9,QF
QRVEXIT:  MOVEM.L   (SP)+,D0-D2
          RTS
QRV010:   CLR.B     QF
          TST.B     QM
          BEQ       QRV030
          CLR.B     QM
          MOVE.L    #1,QI(A1)
QRV020:   MOVE.L    QI(A1),QIND
          JSR       QINDLES
          TST.W     QR(A2)
          BEQ       QRVEXIT
          MOVE.W    QR(A2),D0
          EXT.L     D0
          MOVE.L    D0,QI(A1)
          BRA       QRV020
QRV030:   MOVE.L    QI(A1),QIND
          JSR       QINDLES
          TST.W     QL(A2)
          BEQ       QRV040
          MOVE.W    QL(A2),D0
          EXT.L     D0
          MOVE.L    D0,QI(A1)
          BRA       QRV020
QRV040:   TST.W     QV(A2)
          BNE       QRV050
          CLR.L     QI(A1)
          MOVE.B    #9,QF
          BRA       QRVEXIT
QRV050:   MOVE.L    QKEYLEN(A1),D1
          MOVE.L    A2,D2
          ADD.L     #QY,D2
          TRANSFER  D2,#QZ,D1
          MOVE.W    QV(A2),D0
          EXT.L     D0
          MOVE.L    D0,QI(A1)
          MOVE.L    D0,QIND
          JSR       QINDLES
          MOVE.L    QKEYLEN(A1),D0
          MOVE.L    A2,D1
          ADD.L     #QY,D1
          MOVE.L    #QZ,D2
          COMPARE   D2,D1,D0
          BHI       QRV040          ;QY > QZ
          BRA       QRVEXIT


QEQ_HI:   TST.L     QX(A1)
          BNE       QH010
          MOVE.B    #9,QF
          RTS
QH010:    CLR.B     QM
          JSR       QSEARCH
          MOVEM.L   D0-D2,-(SP)
          MOVE.L    QKEYLEN(A1),D0
          MOVE.L    A2,D1
          ADD.L     #QY,D1
          MOVE.L    #QA,D2
          COMPARE   D1,D2,D0
          MOVEM.L   (SP)+,D0-D2
          BHI       QREAD           ;QA > QY
          RTS


QEQ_LO:   TST.L     QX(A1)
          BNE       QL010
          MOVE.B    #9,QF
          RTS
QL010:    CLR.B     QM
          JSR       QSEARCH
          MOVEM.L   D0-D2,-(SP)
          MOVE.L    QKEYLEN(A1),D0
          MOVE.L    A2,D1
          ADD.L     #QY,D1
          MOVE.L    #QA,D2
          COMPARE   D2,D1,D0
          MOVEM.L   (SP)+,D0-D2
          BHI       QREAD_R         ;QY > QA
          RTS
;
ISAMEND:  EQU       *
```

Ein erster Schritt in FORTH
– eine mehrteilige Einführungsserie

Teil 1

FORTH – warum eigentlich nicht? Schließlich wird FORTH oft mit „richtigen“ Programmiersprachen wie PASCAL, C oder MODULA-2 in einem Atemzug genannt. Und das zu recht: FORTH bietet einiges, was man in diesen Sprachen vergeblich sucht. Daher sollte die Kenntnis der Grundzüge von FORTH jedem, der sich ernsthaft mit Softwareentwicklung beschäftigt, zur Pflichtaufgabe gemacht werden. Denn FORTH ist mehr als nur eine weitere Programmiersprache: Eine eigene Programmierphilosophie.

Ihre prägenden Merkmale sind strukturiertes Design, Modularität, Einfachheit und Erweiterbarkeit. Man könnte es auch auf eine provokante Kurzformel bringen: FORTH Kann alles, was andere Sprachen auch können – meistens allerdings leichter. Die FORTH-Einführungsserie, die in dieser Ausgabe beginnt, will diesen Beweis antreten. ST-COMPUTER-Lesern wird dabei der Einstieg besonders leicht gemacht: Die Serie orientiert sich an VOLKSFORTH, das über den PD-Service erhältlich ist. Alles, was Sie tun müssen, ist Diskette Nr. 3 und Nr. 4 anzufordern: Schon kann's losgehen.

Bis dahin noch ein paar einführende Worte über die Entstehung und die wichtigsten Eigenschaften von Forth.

Wie alles begann

FORTH ist eine Sprache mit bewegter Vergangenheit. Sie entstand Ende der sechziger Jahre und wurde zunächst fast ausschließlich zur Steuerung von Radioteleskopen und zur indusdtriellen Prozeßsteuerung eingesetzt. Jahrelang blieb sie exklusives Programmierwerkzeug einiger weniger „Eingeweihter“. Erst als sich in den siebziger Jahren die Forth Interest Group (FIG) gründete, die mit FIG-FORTH 1.0 (manchmal auch als FORTH-78 bezeichnet) eine Art Standard entwickelte und als Public-Domain zur Verfügung stellte, wurde FORTH auch in der Mikrocomputer-Szene heimisch. Mittlerweile ist für nahezu jeden denkbaren Prozessortyp eine FORTH-Implementation erhältlich – eine Tatsache, die FORTH besonders für Systemprogrammierer interessant macht. Obwohl von „Insidern“ und einigen FAchredakteuren schon seit Jahren eine ständig wachsende Popularität beobachtet wird, hat es FORTH nicht geschafft, sich als „Massensprache“ wie C oder TURBO PASCAL zu etablieren. Die Ursachen dafür sind vielschichtig und sollen an dieser Stelle auch nicht weiter erläutert werden. An der mangelnden Leistungsfähigkeit kann es sicher nicht liegen, denn FORTH weist Eigenschaften auf, die es von Sprachen wie C oder PASCAL Positiv unterscheidet.

Fragen Sie doch einmal einen FORTH-Freak, was denn eigentlichd das Besondere an dieser Sprache sei. Wahrscheinlich wird er Sie zunächst verständnislos anstarren und dann irgendetwas von indirekt verknüpftem Code und virtueller Maschine faseln. Daß sich damit jemand, der von FORTH keine Ahnung hat, nur schwer begeistern läßt, ist sonnenklar. Deshalb will ich versuchen, die wichtigsten Vor- und auch Nachteile von FORTH aufzulisten:

- FORTH ist interaktiv, weist aber gleichzeitig auch Compilereigenschaften auf. Eingegebener Quelltext wird in eine Art Zwischencode übersetzt. Aufgrund der Interaktivität auf der einen und der hohen Compilationsgeschwindigkeit auf der anderen Seite reduziert sich die Programmentwicklungszeit unter FORTH auf ein Minimum.
- FORTH vereinigt Elemente einer Hochsprache mit Elementen einer assemblernahen Sprache. So kann z. B. Maschinencode beliebig mit Hochsprache kombiniert werden, was u. U. zu einer weiteren Geschwindigkeitssteigerung führt.
- Zwar bietet FORTH dem Benutzer standardmäßig nur sehr wenige Datentypen, es erlaubt jedoch die Definition beliebiger Datentypen.
- Ein FORTH-System kommt mit geringem Speicherplatz aus. So kann ein Minimalsystem durchaus in 2 oder 3 KByte untergebracht werden. Dies mag zwar angesichts eines 1-MByte-RAM-Speichers zunächst uninteressant sein, spielt aber dann eine Rolle, wenn der ST die Rolle eines Entwicklungssystems spielt und man z. B. mit Hilfe eines sog. „Metacompilers“ Applikationen etwa für Einplatinensysteme erstellen möchte.
- Zwar ist der FORTH-Compiler nur mit minimalem Komfort ausgestattet (so wird z. B. weder die Verträglichkeit von Datentypen überprüft, noch wird sichergestellt, daß Benutzerdaten nicht mit Programmdaten kollidieren), dafür sind FORTH-Programme in der Regel recht schnell (siehe auch Abb. 3).
- FORTH ist weitgehend standardisiert. Als anerkannter Standard hat sich mittlerweile der FORTH-83 Standard etabliert. Die meisten FORTH-Programme orientieren sich an diesem Quasi-Standard.

Infokasten 1:
FORTH-Steckbrief

Infokasten 2:
Der FORTH-Standard

Infokasten 3:
Wie schnell ist FORTH?

Infokasten 4:
Welche FORTH-Systeme gibt es für den ST?

INFOKASTEN:

Welche FORTH-Systeme gibt es für den ST?

Für den ATARI ST sind mittlerweile mehr als ein halbes Dutzend FORTH-Versionen erhältlich. Alle diese Versionen orientieren sich am FORTH-83-Standard. Leider gilt dieser Standard natürlich nur für die hardwareunabhängigen Worte des Sprachkerns. Beim Rest (und der macht in der Regel den größeren Teil des Sprachkerns aus, bleibt es dem Entwickler überlassen, wie er die entsprechenden Worte benennt. Glücklicherweise hat sich auch hier, zumindest beim Aufruf der Betriebssystemroutinen des ST, eine relativ einheitliche Syntax durchgesetzt. Die auf dem Markt angebotenen Systeme unterscheiden sich natürlich in erster Linie durch ihren Preis. Am oberen Ende rangieren Systeme wie 4xFORTH, die etwa soviel kosten wie ein guter C-Compiler. Am unteren Ende der Preisskala rangieren Systeme wie ST_FORTH oder das VOLKSFORTH. Die Vertreiber von 4xFORTH etwa bieten dem Kunden sehr viel, was über die reine Software hinausgeht u. a. etwa eine erstklassige technische Unterstützung, kostenlose Updates und eine User Zeitschrift. Die Preisunterschiede haben ihren Grund also nicht unbedingt in unterschiedlicher Qualität.

Für den Einstieg ist ein System wie VOLKSFORTH bestens geeignet. Es bietet alles, was zur Programmentwicklung notwendig ist. Beim über den PD Service angebotenen VOLKSFORTH Handelt es sich um Version 3.7. Sie wurde inzwischen von dem Autor überarbeitet. Bei der aktuellen Version 3.8 wurde u. a. das File Interface verbessert und ein vollständig in GEM eingebundener Editor integriert. Version 3.8 ist zusammen mit einer umfangreichen Dokumentation für 60 DM bei der FORTH-Gesellschaft in Hamburg (einer nicht kommerziellen Vereinigung von FORTH-Programmierern) erhältlich. Wer bereits Version 3.7 besitzt erhält nach Einsendung von 20 DM und drei leeren Disketten ein Update. Adresse: FORTH-Gesellschaft e.V. Friedensallee 92, 2000 Hamburg 50 Tel. 0 40-390 42 04

Programmentwicklung unter FORTH

Wer sich für eine neue Programmiersprache interessiert, wird wohl zunächst seinen Blick auf den Aufbau eines typischen Programmes richten. Die wohl markanteste Eigenschft von FORTH ist seine Erweiterbarkeit: Ein FORTH-System (so wie Sie es z. B. über den PD-Service erhalten können) besteht u. a. aus einer bestimmten Anzahl von Worten (stellen Sie sich unter einem Wort zunächst einfach eine Procedure wie in Turbo Pascal oder noch einfacher ein Unterprogramm wie in BASIC vor), dem Sprachkern des Systems. Die Gesamtheit aller Worte eines Systems ist in einem Teil des Arbeitsspeichers untergebracht, der als Wörterbuch bezeichnet wird. Programmieren in FORTH heißt nun, neue Worte zu definieren. Jedes neu definierte Wort baut zwangsläufig auf bereits existierenden Worten auf. Ein einfaches Beispiel soll dies erläutern. Nehmen wir an, Sie sollten ein Programm für die Steuerung einer Verkehrsampel erstellen. Sinnvollerweise beginnt man dabei zunächst mit der Steuerung einer einzigen Ampel, um über die grundsätzliche Vorgehensweise Klarheit zu gewinnen.

Zunächst werden Worte benötigt, die die drei Lampen Rot, Gelb und Grün steuern. Genauer gesagt: Bei Aufruf eines dieser Worte soll eine der drei Lampen ein- bzw. ausgeschaltet werden. In FORTH würden diese Worte folgendermaßen aussehen:

```
: ROT_EIN SET PORT1 ON ;
: ROT_AUS SET PORT1 OFF ;
: GELB_EIN SET PORT2 ON ;
: GELB_AUS SET PORT2 OFF ;
: GRÜN_EIN SET PORT3 ON ;
: GRÜN_AUS SET PORT3 OFF ;
```

Hier wird der charakteristische Aufbau eines FORTH-Wortes sichtbar. Es wird durch ':' eingeleitet, gefolgt von dem Namen, unter dem das Wort aufgerufen werden kann. Nun folgen die Worte, die bei Aufruf des gerade definierten Wortes ausgeführt werden sollen. Beendet wird die Wortdefinition durch ein Semikolon. Beim Aufruf etwa von ROT_EIN werden der Reihe nach die Worte 'SET', 'PORT1' 'und 'ON' ausgeführt und bewirken, daß die rote Lampe eingeschaltet wird. (In der Praxis könnte man z. B. die Lampen an die serielle Schnittstelle des ST anschließen. 'SET PORT1 ON' würde dann einen Ausgang der Schnittstelle auf '1' setzen. Machen Sie sich über die Worte 'SET', 'PORT' und 'OFF' keine weiteren Gedanken. Es sind keine Worte des FORTH-Kerns, sondern sie wurden zuvor vom Benutzer definiert. Sie dienen hier lediglich zur Veranschaulichung.) Entsprechend verhalten sich die übrigen fünf Worte.

Legen Sie an dieser Stelle einmal eine kurze Pause ein und notieren Sie sich einen ebenso trivialen wie bedeutungsvollen Satz: Das Verhalten eines FORTH-Wortes bei seiner Ausführung wird durch das Verhalten seiner Komponentenwörter bestimmt. (Damit sind jene Wörter gemeint, die das jeweilige Wort beinhaltet.)

Unsere sechs Beispielworte sind also wiederum in FORTH geschrieben. Überhaupt ist ein großer Teil dieser Sprache durch sich selbst definiert. Dies kann natürlich nicht beliebig fortgesetzt werden, denn keine Sprache kann vollständig in sich selbst definiert sein. Irgendwann wird ein FORTH-Wort nicht mehr aus weiteren FORTH-Worten, sondern direkt aus Maschinencode bestehen. Solche Wörter werden im weiteren Verlauf der Serie noch des öfteren auftauchen.

Zurück zu unserer Ampelsteuerung. Mit den sechs Worten zur Ansteuerung ist es alleine noch nicht getan. Skizzieren Sie sich einmal den Ablauf einer Ampelphase. Die einzelnen Phasen lauten: ROT – ROT/GELB – GRÜN – GELB. Dabei fällt zunächst auf, daß vorhin ein Wort vergessen wurde, dessen Definition an dieser Stelle nachgeholt werden soll:

```
: ROT/GELB_AUS
GELB_AUS ROT_AUS ;
```

So einfach ist das. Anstatt die Worte 'GELB_AUS' und 'ROT_AUS' der Reihe nach aufrufen zu müssen, wird durch das Wort 'ROT/GELB_AUS' das gleiche erreicht. Wenn Sie jetzt allerdings auf die Idee kommen sollten, zu sagen „Alles klar" und ein Wort hinzuschreiben, das etwa wie folgt aussehen könnte:

```
: AMPEL ROT_EIN
GELB_EIN
ROT/GELB_AUS
GRÜN_EIN
GRÜN_AUS
GELB_EIN
GELB_AUS ;
```

und den Ablauf einer kompletten Phase beschreibt, so wäre dies nur fast richtig. Außer acht gelassen wurde bislang die Tatsache, daß zwischen den einzelnen Phasen natürlich auch Wartephasen liegen müssen. Da sie verschieden lang sind, wäre es denkbar, für jede Wartephase ein eigenes Wort zu definieren. Wesentlich flexibler wäre es allerdings, wenn man ein einziges Wort definieren würde, bei dem die jeweilige Wartezeit vor dem Aufruf als Parameter übergeben wird. Damit steht auch gleich ein weiterer Punkt an, von dem bislang noch nicht die Rede war – die Parameterübergabe. In Hochsprachen wie C oder PASCAL werden die einer Prozedur zu übergebenden Parameter in der Regel als lokale Variablen explizit definiert und vor jedem Prozeduraufruf übergeben. In FORTH geht dies viel einfacher. Hier wird generell ein Ort vereinbart, an dem vor dem Aufruf eines Wortes alle benötigten Parameter abgelegt werden. Bei diesem Ort handelt es sich um einen bestimmten Bereich des Arbeitsspeichers, der speziell für diese Bestimmung reserviert ist und als Parameterstack bezeichnet wird. Für den Zugriff auf diesen „Speicherbereich für lokale Variablen" gilt das Prinzip „Was zuletzt abgelegt wurde, wird als erstes wieder heruntergenommen". Ich werde in der nächsten Folge dieses einfache Prinzip an einigen Beispielen erläutern, damit auch die Funktion des Parameterstacks deutlicher wird. Um eine Zahl im Stack abzulegen, sind keine besonderen Worte nötig: Jede eingegebene Zahl wird grundsätzlich erst einmal im Stack abgelegt. Für den Anfang soll diese kurze Beschreibung genügen.

Mit diesem Wissen kann nun auch ein Wort definiert werden, das eine bestimmte Zeit wartet. Die genaue Dauer des Wartezustandes hängt von der Größe der Zahl ab, die zuvor auf dem Parameterstack abgelegt wurde.

```
: WARTE O DO LOOP ;
```

Innerhalb von 'WARTE' läuft eine einfache Schleife vom Typ 'DO/LOOP', die von Null bis zu dem übergebenen Wert zählt. Damit sind alle Bausteine beisammen, die für die eingangs gestellte Aufgabe benötigt werden. Das komplette Wort könnte folgendes Aussehen haben:

```
: AMPEL
  ROT/GELB/GRÜN_AUS
  BEGIN
    ROT_EIN 1000 WARTE
    GELB_EIN 200 WARTE
    ROT/GELB_AUS
    GRÜN_EIN 1000 WARTE
    GRÜN_AUS
    GELB_EIN 200 WARTE
    GELB_AUS
  AGAIN 1
```

In das Wort 'AMPEL' wurde eine Wiederholungsanweisung vom Typ 'BEGIN/AGAIN' eingebaut, welche bewirkt, daß alle Anweisungen, die zwischen 'BEGIN' und 'AGAIN' auftauchen, theoretisch unendlich oft durchlaufen werden.

Damit wäre der Quellcode für unser erstes FORTH-Programm fertig, allerdings bislang nur auf dem Papier. Um den Quellcode zur Ausführung zu bringen, können wir ihn entweder über die Tastatur eintippen. Der FORTH-Interpreter würde die gesamte Eingabe analysieren und das Wörterbuch um die neuen Wörter erweitern. Anschließend kann das Wort 'AMPEL' (genau wie jedes seiner Komponentenwörter) durch Eingabe des Wortnamens zur Ausführung gebracht werden. Der Nachteil dieses Verfahrens liegt auf der Hand – ein auf diese Weise eingegebener Quelltext kann nicht mehr modifiziert werden. Die Alternative wäre, den Quellcode zunächst mit Hilfe eines eingebauten Editors einzugeben und auf einem Massenspeicher (Diskettenlaufwerk oder Festplatte) abzuspeichern. Von dort kann er dann bei Bedarf geladen werden. Dieses „Laden" von Diskette (das in FORTH zuständige Wort heißt 'LOAD' erzielt exakt den gleichen Effekt wie die direkte Eingabe über Tastatur. Der FORTH-Interpreter behandelt von der Tastatur kommende Daten genau wie die von einer Massenspeichereinheit kommenden. Dieses im Grunde einfache aber dennoch leistungsfähige Konzept erlaubt dem FORTH-Programmierer, sich (ähnlich wie etwa in C) Bibliotheksmodule anzulegen, die bei Bedarf zum System hinzugeladen werden.

Damit soll die Übersicht für das erste beendet sein. Sinn dieses Beispiels war es nicht, einen Schnellkurs für FORTH in zehn Minuten durchzuführen, sondern, an einem anschaulichen Beispiel einige charakteristische Merkmale der Programmierung unter FORTH vorzuführen. Programmieren in FORTH heißt nicht etwa nur das Ineinanderschachteln einzelner Programmodule, denn dann könnte man besser bei BASIC, PASCAL oder vielleicht auch bei LOGO bleiben. Programmieren in FORTH heißt vielmehr, ein Problem auf die einfachste und eleganteste Weise zu lösen.

Zum Schluß noch ein Aspekt, der fairerweise auch einmal angesprochen werden sollte. Wie sieht es mit der Erlernbarkeit von FORTH aus? FORTH macht auf Neulinge, die Sprachen wie Turbo Pascal, BASIC oder gar FORTRAN gewöhnt waren einen eher merkwürdigen Eindruck. Dazu trägt nicht zuletzt die Verwendung der Umgekehrt Polnischen Notation zur Darstellung arithmetischer Ausdrücke und eine recht unkonventionelle Syntax bei. C-Programmierer haben es am Anfang ein wenig leichter, da auch in FORTH viele Befehle durch Bezeichner dargestellt werden, die aus einem oder zwei Zeichen bestehen. Auch hier gilt die Devise „Lernen durch Ausprobieren". Deswegen möchte ich Sie ermutigen, in dieser Einführungsserie auch weiter am Ball zu bleiben. Ich hoffe, bereits mit dieser Einleitung Vorfreude geweckt zu haben. Bis zum nächsten Mal – und vergessen Sie nicht, sich eine FORTH-Version aus dem PD-Service von ST-Computer zu besorgen!

Peter Momadjemi

INFOKASTEN:

Wie schnell ist FORTH?

Über den Sinn und Unsinn von einfachen Benchmarktests ist schon viel geschrieben worden. Auch wenn man diese Tests nicht überbewerten sollte, geben sie doch einen ersten Aufschluß über die Leistungsfähigkeit eines Systems. Als Standard hat sich das Sieb des Eratosthenes entwickelt (Listing in Heft 11/86). In der nachfolgenden Tabelle wurde die Ausführungsgeschwindigkeit der gängigsten Programmiersysteme gegenübergestellt.



SALIX PROLOG

FÜR ALLE ATARI ST

mit ausführlichem deutschem Handbuch

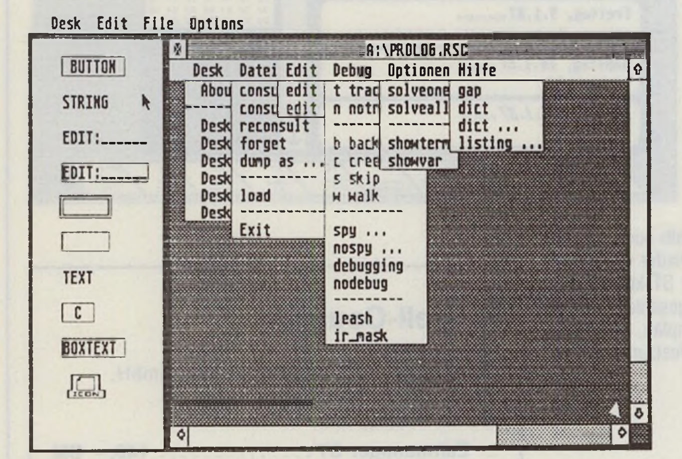
Darüberhinaus wurde SALIX PROLOG mit weiteren nützlichen Funktionen versehen. Insgesamt werden ca. 140 eingebaute Funktionen („Primitives") unterstützt. Außerdem wird eine Bibliothek mitgeliefert, die es gestattet, fast alle Funktionen des GEM als PROLOG-Funktionen zu benutzen. SALIX PROLOG hat eine komfortable GEM-Oberfläche mit Drop-Down Menüs und Formularen und einem Window-Management-System. Diese GEM-Oberfläche wurde selbst in SALIX PROLOG geschrieben.

```
Desk Datei Edit Debug Optionen Hilfe          158278 Bytes frei
     consult user   consult user
     consult      - Top Level Interpreter          debug & help
     reconsult
     forget                                  fractaltree(H) :-
     dump as ...                                  H>3,
     ----------                                   wind_get(_,10,Wind,_,
     load                                         wind_get(Wind,4,_,_,_,
     ----------                                   H1 is 0.1199999999999
     Exit                                         H2 is H/ 4,
                                                  fractaltree(H,L,H2),
                                                  graf_mouse(off),
                                                  fractal([H1|L]),
                                                  graf_mouse(on) .
                                             fractaltree(1,[],_) :-
                                                  ! .
                                             fractaltree(N,[H|L],H) :-
                                                  N1 is N-1,
                                                  fractaltree(N1,L,H) .
```

Leistungsdaten:

— *Compiler/Interpreter System.* Der inkrementelle Compiler erzeugt einen Zwischencode, der vom Interpreter abgearbeitet wird.

- **Edinburg Standard**
  Alle im Buch „PROLOG — Programming in Logic" von Clocksin & Mellish definierten Funktionen sind verfügbar.
- **zusätzliche eingebaute Funktionen**
  (primitives), insgesamt ca. 140 Funktionen
- **Integer- und Gleitkommaarithmetik**
  mathematische Funktionen, Gleitkomma in doppelter Genauigkeit.
- **Benutzerausgänge:**
  — ***Exception Handler.*** Bei einem Fehler kann der Benutzer mit dem Prädikat 'error' eigene Fehlerroutinen in Kraft setzen.
  — ***Leistungsfähiges Testsystem.*** Das Testsystem ist selbst in PROLOG geschrieben und kann bei Bedarf modifiziert werden.
  — ***portray-Prädikat.*** Das Standardausgabe-Prädikat 'print' kann vom Benutzer modifiziert werden.
  — ***Einbindung benutzergeschriebener Precompiler.*** Die Einbindung erfolgt in die Prädikate 'consult' und 'reconsult'. Beispielsweise können so Bibliotheken, die aus Grammatikregeln oder Regeln für ein Expertensystem bestehen, direkt konsultiert werden.
  — ***Benutzerausgang bei Zeicheneingabe.*** Damit können komfortable Benutzeroberflächen für Expertensysteme angeschlossen werden.
- **Syntax Error Corrector**
  Tritt beim Einlesen eines PROLOG-Ausdrucks (vom Bildschirm oder vom Laufwerk) ein Syntaxfehler auf, erhält der Benutzer Gelegenheit, den Fehler am Bildschirm zu verbessern.
- **Lector.** Ein in PROLOG geschriebenes Hilfsmittel, um PROLOG-Programme auf Tippfehler und falsche Verwendung der eingebauten Funktionen zu prüfen.
- **Anschluß an einen vorhandenen Editor.** (z.B. 1st-Word) Die PROLOG-Sitzung muß nicht verlassen werden, um eine Bibliothek zu editieren.
- **GEM-Library** gestattet die Nutzung fast aller GEM Funktionen von PROLOG aus, damit steht die riesige Grafikbibliothek des GEM in der Form von PROLOG-Prädikaten zur Verfügung.
  Zusätzlich Window Management System mit Pufferung und automatischem Refresh der Fenstertextinhalte.



- **GEM-Top-Level-Interpreter,** komfortable Bedienung des PROLOG-Programmiersystems mit Drop-Down-Menüs und Formularen. Der Gem-Top Level Interpreter ist komplett in PROLOG geschrieben, damit auch für spezielle Anforderungen modifizierbar.
- **Startup-File,** um das PROLOG-System entsprechend den jeweiligen Anforderungen zu konfigurieren.
- Der **volle Adressraum des 68000** wird unterstützt, so volle Ausnutzung des ATARI-Speichers und etwaiger Speichererweiterungen.
- **Automatische Garbage Collection.**
- **Beispiel- und Utility-Bibliotheken.**

DM **198,-**

Handbuch einzeln 60,— wird beim Kauf angerechnet

Heim-Verlag
Heidelberger Landstr. 194
6100 Darmstadt-Eberstadt
Telefon: (0 61 51) 5 60 57

WO BEKOMME ICH SALIX PROLOG?

Ein Expertensystem gibt Antwort

Computer: Welches Problem haben Sie?
Leser: Ich suche ein Programm, mit dem ich Probleme der künstlichen Intelligenz lösen kann.
Computer: Wollen Sie auch Expertensysteme, etwa so wie mich programmieren?
Leser: Was sind Expertensysteme?
Computer: Programme wie ich, wo Ihnen der Computer mit Fachwissen ein Problem lösen kann.
Leser: Ja, das ist wohl eine Hauptanwendung.
Computer: Soll dieses Programm auf einem leistungsfähigen modernen 68000 Computer laufen?
Leser: Selbstverständlich, denn die Sprache soll ja schnell sein.
Computer: Soll das Programm auch unter einer benutzerfreundlichen Oberfläche, etwa GEM laufen?
Leser: Das wird wohl optimal sein.
Computer: Wie soll das Preis-Leistungsverhältnis sein?
Leser: Möglichst günstig.
Computer: Da kann ich Ihnen nur das Salix Prolog für den Atari St empfehlen.
Leser: Ist das denn nicht zu teuer?
Computer: Nein, trotz eines guten Leistungsverhältnisses kostet es nur 198.— DM.
Leser: Das ist ja toll. Wo kann ich es bekommen?
Computer: Das ist ganz einfach. Untenstehenden Coupon ausfüllen und an den Heim Verlag, Heidelberger Landstr. 194, 6100 Darmstadt einsenden.

BESTELL-COUPON

Einsenden an: Heim Verlag, 6100 Da.-Eberstadt, Heidelberger Landstr. 194

Bitte senden Sie mir:
— St. SALIX-PROLOG mit über 250 S. deutschem Handbuch
zum Preis von 198,— je Stück.
zzgl. DM 5,— Versandkosten bei NN
☐ per Nachnahme ☐ Verrechnungsscheck liegt bei

Name
Straße
Ort


hardware
software
organisation
service
CSF GMBH
Heeper Str. 106–108, 4800 Bielefeld 1, 0521/61663
Kein Kabelsalat mehr mit dem Gehäuse für ATARI ST
● Zentrale Stromversorgung für alle Geräte einschl. 2 Drucker
● Einbaumöglichkeit von 2 Diskettenlaufwerken
● Rechner (Tastatur) kann komplett unter das Gehäuse geschoben werden (Staubschutz)
● Massives Blechgehäuse
ATARI ST-Gehäuse erhalten Sie bei den autorisierten Fachhändlern
DM 198,–
NEUES VON CSF – Speichererweiterung ohne Probleme!
– steckbar (ohne jegliche Lötarbeiten) – läuft auch auf dem 520 STM
– enorme Zeitersparnis durch einfache, bebilderte Einbauanleitung
– kein Flimmern nach der Erweiterung (durch separate, geglättete Spannung an der zweiten RAM-Bank)
– sensationeller Preis
– Bei Bestellungen bitte angeben: ☐ Speichererweiterung steckbar ☐ Speichererweiterung zum Löten
41256
SHIFTER
CSF
256k x 16
DM 225,–
Zu beziehen:
Direkt bei CSF, Bielefeld Tel. 0521/61663
Bei allen ATARI-Händlern
In der Schweiz: SENN Computer AG Langstr. 31 · CH-8021 Zürich Tel. 01-2417373
In Österreich: Institut für Datenverarbeitung und Organ. Ges.mbH Rehberger Hauptstr. 95 · A-3503 Krems Tel. 02732-70581
Alle Preise sind unverb. empf. Verkaufspreise

ST-Ecke

Listing 1

```
/****************************/
/*    ptadr()-Routine      */
/*        in C             */
/*      von S.H.           */
/****************************/
char *pt_adr(tree,ind,clear)
long tree;                          /* Baumadresse */
int ind;                            /* Objektindex */
{
    OBJECT *obj;
    TEDINFO *ted;
    char *text_adr;

    obj=(OBJECT*)(tree+24l*ind);   /* Baumadresse + Objektindex*24 */
    ted=(TEDINFO*)(obj->ob_spec);  /* Inhalt von Objektadresse + ob_spec(12)*/
                                    /* ob_spec-Inhalt zeigt auf TEDINFO-Eintrag */
                                    /* des Textes */
    text_adr=ted->te_ptext;         /* TEDINFO-Eintrag PTEXT zeigt dann auf */
                                    /* den Textstring */
    if (clear)                      /* Wenn dieses Flag gesetzt ist, dann */
      *text_adr=0;                  /* wird der erste Buchstabe auf Null */
                                    /* gesetzt. Dadurch steht der Cursor am */
                                    /* Anfang.*/
    return(text_adr);               /* Adresse zurückgeben. */
}

Oder etwas kürzer:

char *pt_adr(tree,ind)
long tree;
int ind;
{
 return((char*)(((TEDINFO*)(((OBJECT*)(tree+24l*ind))->ob_spec))->te_ptext));
}
```

Listing 2

```
' **********************************
' *                                *
' * Pt_adr-Routine in GFA-Basic    *
' *                                *
' *     HE            3.7.87       *
' *                                *
' **********************************
'
Procedure Pt_adr(Tree%,Ind%,Clear%,Txt_adresse%)
  '
  Te_ptext=0
  Te_ptmplt=4
  Te_pvalid=8
  '
  Obj_adresse%=Tree%+Ind%*24                       ! Baumadr. + Ind * 24
  Tedinfo_adresse%=Lpeek(Obj_adresse%+12)          ! Inhalt Obj.adr + ob_spec(12)
  Txt_adresse%=Lpeek(Tedinfo_adresse%+Te_ptext)    ! Textanf. = TEDINFO-ptext
  If Clear%<>0 Then
    Poke Txt_adresse%,0                            ! Null setzen
  Endif
  *Txt_adresse=Txt_adresse%                        ! Rückgabe Textanfangsadr.
Return
```

Liebe Leserinnen und Leser,

ab dieser Ausgabe wollen wir uns häufig gestellten Fragen, die sich mit dem Programmieren auf dem ATARI ST beschäftigen, widmen. Dabei soll nicht auf die Hardware, die Handhabung von Anwenderprogrammen oder das Erstellen von Druckeranpassungen u. a. eingegangen werden. Vielmehr sollen hier kleinere Probleme des täglichen Programmierens angesprochen und erklärt werden. Dazu ist natürlich Ihre rege Beteiligung Ihrerseits erforderlich. Nur wenn Sie mit Ihren Fragen an uns herantreten, können wir wissen, welche Schwierigkeiten am häufigsten beim Umgang mit dem ATARI ST auftreten. Wir wollen in dieser Kolumne auch Fehler des Gerätes ansprechen und versuchen, Ihnen beim Beheben oder Umgehen dieser Fehler behilflich zu sein.

Bitte richten Sie Ihre Fragen an folgende Adresse:

ST-Computer-Redaktion
STICHWORT ST-ECKE
Schwalbacher Str. 64
6236 Eschborn

Michael Schachenmayr aus Weilheim stellte folgende Frage:

Wie bearbeitet man die in einem Objektbaum eingegebenen Texte am sinnvollsten? Kann man einen Objektbaum mit mehreren Textfeldern mit dem RCS so erstellen, daß der Cursor später am Anfang der editierbaren Felder erscheint und nicht am Ende? Es ist ziemlich anwenderfeindlich, immer vorher ESC zu drücken! Wie bekommt man die Adresse eines Textfeldes heraus?

Von den GEM-Schreibern wurde der '@' als sogenannter META-Buchstabe eingeführt: Würde man ihn bei einem Text an den Anfang stellen, so verschwindet der nachfolgende Text. Man muß, um für später Speicherplatz zu reservieren, im RCS bei PTEXT einen Text in der Größe des Textfeldes ein-

geben! Schreibt man nun im RCS vor diesen Text den Buchstaben '@ 'so verschwindet der nachfolgende Text schon im RCS. Leider ist diese Möglichkeit also mit dem RCS nicht möglich. Wenn wir aber im Programm nach dem Laden des Resource-Files die Inhalte der Texte durch Nullstrings ersetzen (der erste Buchstabe hat den Wert 0), so steht im Speicher kein Text mehr für das Textfeld, d. h. der Cursor erscheint am Anfang. Den Nachteil will ich nicht verschweigen: Man sollte jedes Textfeld am Anfang auf Null setzen. Wenn man viele Dialogboxen mit Textfeldern besitzt ist das natürlich sehr arbeitsaufwendig. Ist es nötig, diese Textfelder öfters zu löschen, so könnte man sich ein Feld anlegen, in dem alle zu bearbeitenden Textfelder enthalten sind. Diese kann man dann mit der untenstehenden Funktion bearbeiten.

Wie errechnet man nun die Adresse eines Textfeldes? Dabei ist die Verknüpfung der Zeiger folgendermaßen:

- Adresse des Baumes errechnen. Dies macht man am besten am Anfang des Programmes.
- Objektindex ★ 24 (Länge der Objektstruktur) zur Baumadresse addieren. Damit erhält man den Anfang des entsprechenden Objektes. Addiert man nun 12 dazu, erhält man die Adresse von ob_spec. Darin steht ein Zeiger, der auf TEDINFO zeigt.
- An dieser Adresse steht dann nun die Adresse des Strings.
- Nun muß gegebenenfalls nur noch an diese Adresse eine Null hineingeschrieben werden.

Ich habe zum besseren Verständnis ein Beispiel in C und in GFA-BASIC 2.0 beigefügt.

Der Aufruf würde in C folgendermaßen aussehen:

Errechnen der Adresse des Textfeldes von Objekt 4 bei vorhandener Baumadresse in der Variablen 'baum'

ohne Löschen des Feldes...

Adresse=pt_adr(baum,4,0);

mit Löschen des Feldes...

Adresse=pt_adr(baum,4,1);

Listing 3

```
Programmbeispiel in C:

/*   Laden eines Bildes mit Betriebssystemroutinen */

#include <osbind.h>
main()
{
    int fd;
    long pic_adr;
    pic_adr=Logbase();                      /* Adresse des Bildschirmspeichers */
    fd=Fopen("beispiel.pic",0);            /* Datei anlegen */
    if (fd<=0)
    {
        /* Fehlermeldung ausgeben */
    }
    Fread(fd,32000l,pic_adr);               /* Bildschirm speichern */
    Fclose(fd);                             /* Datei schließen */
}

/* Speichern eines Bildes mit Betriebssystemroutinen */

#include <osbind.h>
main()
{
    int fd;
    long pic_adr;

    pic_adr=Logbase();                      /* Adresse des Bildschirmspeichers */
    fd=Fcreate("beispiel.pic",0);           /* Datei anlegen */
    if (fd<=0)
    {
        /* Fehlermeldung ausgeben */
    }
    Fwrite(fd,32000l,pic_adr);              /* Bildschirm speichern */
    Fclose(fd);                             /* Datei schließen */
}
```

Die GFA-Basic-Routine kann man Listing 2 entnehmen.

Aufruf der Adresse mit Löschen:
Gosub Pt_adr(Baum%,4,1,★Txt_adr%)

Aufruf der Adresse ohne Löschen:
Gosub Pt_adr(Baum%,4,0,★Txt_adr%)

Eine Umsetzung in PASCAL oder ASSEMBLER müßte durch die beiden Beispiele nun einfach durchzuführen sein.

Eine weitere häufige Frage: Wie kann man den Grafikbildschirm auf Diskette abspeichern?

Dazu läßt man sich vom Betriebssystem mit der Funktion xbios(3) die Anfangsadrese des Bildschirms mitteilen und speichert ab dieser Stelle 32000 Bytes ab. Sollten Sie Farbbilder abspeichern wollen, so müssen Sie die Farbregister abspeichern und diese beim Laden des Bildes auch wieder setzen.

Das Programmbeispiel in C kann man Listing 3 entnehmen.

GFA Basic-Beispiel:

Laden des Bildes:
Bload "beispiel.pic",Xbios(3)

Speichern des Bildes:
Bsave "beispiel.pic",Xbios(3),32000

Einfach nicht?

Nächsten Monat werde ich näher darauf eingehen, wie man die Farbpalette aus den Registern liest.

(Stefan Höhn)



Alle Macht den Tasten

Formatierte Eingaberoutine für PASCAL

Mancher PASCAL-Programmierer kennt das Problem der Standard-Eingabefunktionen READ und READLN. Will man eine Zahl einiesen und tippe versehentlich eine Buchstabentaste, ist das Programm unzufrieden. Die folgende Routine läßt dieses Manko vergessen, bietet ganz nebenbei noch eine Formatvorgabe und begrenzt auf Wunsch die erlaubten Tasten.

Wer die Dialogboxen von GEM kennt, wird sie sicher zu schätzen wissen. Doch beim Erstellen eigener Programme ist die Handhabung dieser Boxen nicht gerade einfach, vor allem für diejenigen, die sich noch nicht eingehend mit den Resourcen und Objektbäumen beschäftigt haben.
Doch nun zu unserer Routine. Zuerst die Syntax:

INPUT (x,y,maxlaenge,default,erlaubt,formatstring,inputstr)

wobei

x	=	x-Position der Zeile (integer)
y	=	y-Position der Zeile (integer)
maxlaenge	=	Länge des einzugebenden Strings (integer)
default	=	vorbelegt er Text im Eingabestring (string)
erlaubt	=	gültige Eingabetasten (['A'..'Z'], ['a'..'z'], ['0'..'9'] oder auch ['E'..'H']
formatstring	=	Eingabemaske. (string) '_' und '$' = Platzhalter für Zeichen
inputstr	=	Variable mit dem eingegebenen Ergebnis (string)

Die Routine ist unter ST-Pascal von CCD lauffähig und dient als Ergänzung der PASCAL'schen Standard-Eingabebefehle. Sie läßt sich leicht in eigene Programme einbauen oder sogar als Include-Datei per Compileranweisung automatisch einfügen (z. B. {$ I Eingabe }). Dabei sollte man allerdings folgendes beachten: Am Programmanfang müssen einige, zum Funktionieren unbedingt erforderliche Deklarationen erfolgen. Das sind der Typ 'chmenge' und die Boolsche Variable 'ftaste', ohne die die Routine nich tläuft.

Doch nun zu der aus zwei Teilen bestehendenRoutine: Die Funktion 'GetChar' wird von der Hauptroutine aufgerufen, ist aber auch getrennt von ihr gut einsetzbar. Deswegen wurde sie auch nicht lokal zur eigentlichen Inputroutine deklariert.

Sie nutzt einen GEMDOS-Aufruf, um auf einen Tastaturdruck ohne Bildschirmecho zu warten. Dieser Aufruf besteht aus zwei Wörtern (32 bit). Im zweiten Wort wird der ASCII-Code der gedrückten Taste zurückgegeben, im ersten ein besonderer Tastaturcode (SCAN-Code), der dann von Bedeutung ist, wenn eine Funktionstaste oder eine Taste aus dem Pfeilblock gedrückt wurde. In diesem Fall ist der zurückgegebene ASCII-Code nämlich, unabhängig von der betätigten Funktionstaste, immer Null.

Für den Fall, daß eine solche Taste gedrückt wurde, gibt die Routine den Tastaturcode als Char-Wert zurück und setzt zur Unterscheidung von einem ASCII-Code die Variable 'ftaste' auf true. Ansonsten wird 'ftaste' auf false gesetzt, und die Routine gibt als Ergebnis das Zeichen zurück.

Die Prozedur 'input' ist der Hauptteil der Routine. Sie verlangt insgesamt sieben Übergabeparameter. Die ersten drei davon geben die Startposition auf dem Bildschirm sowie die erlaubte Länge der Eingabe an, wobei die Länge 80 Zeichen nicht überschreiten sollte, da ST-Pascal sonst mit einem STRING-OVERFLOW-Error antwortet. Außerdem sollte die Eingabe nur in einer Zeile stattfinden.

Der Parameter 'default' gibt einen Standardstring an, der als Eingabevorgabe eingeblendet wird. Somit kann diese Funktion auch einen momentanen Variableninhalt ändern. Der alte Wert wird über 'default' angezeigt und dann nach Belieben editiert. Dies ist bei einer Dateiverwaltung ein äußerst nützliches Hilfsmittel.

Der alte Wert kann während einer Eingabe durch Drücken von Ctrl-C bzw. UNDO wiederhergestellt werden.

Der fünfte Parameter 'erlaubt' übergibt in einem 'set of char' alle Zeichen, die die Prozedur als Eingabe akzeptiert. Alle anderen Zeichen werden durch ein Signal zurückgewiesen. Dadurch kann man beispielsweise nur Zahlen oder bestimmte Buchstaben zulassen.

Der sechste Parameter 'Formatstring' übergibt der Prozedur einen String, in dem eine Eingabemaske festgelegt wird, in der bestimmte Zeichen nicht überschrieben werden können (ähnlich wie z. B. beim Kontrollfeld die Eingabemasken für Datum und Uhrzeit). Dieser Formatstring kann ruhig ein bißchen schludrig oder als Leerzeichen ('') übergeben werden. Ist die Anzahl der erlaubten Eingabepositionen im Formatstring kleiner als die mit Maxlänge angegebene Anzahl, so fügt das Programm automatisch die entsprechende Anzahl von Hintergrundzeichen ein, und zwar sucht es das erste im Formatstring vorkommende Hintergrundzeichen. Welche Stellen im Formatstring als Hintergrundzeichen interpretiert werden, also überschrieben werden können, wird in der Variable 'editierbar' festgelegt. Diese Variable habe ich hier nicht als Übergabeparameter vorgesehen, dies ist aber ohne weiteres möglich. Der letzte Parameter 'inputstr' liefert den Funktionswert der INPUT-Routine. Dabei werden jedoch nur die tatsächlich eingegebenen Zeichen zurückgegeben, die Formatzeichen aus dem Formatstring werden selbstverständlich ignoriert.

In der Prozedur 'input' sind noch einige weitere Prozeduren enthalten, die kleine Teilaufgaben erfüllen und durch die modulare Schreibweise leicht nachträglich zu ändern sind. So bewegt die Prozedur 'GoPos' den Cursor an die aktuelle Cursorposition (in der Variable 'curpos' abgespeichert). Die Prozedur 'Ausgabe' baut die Eingabemaske samt der bisher durchgeführten Eingabe neu auf, was nach jeder akzeptierten Eingabe, die den String verändert, geschieht. Dies ist notwendig, da der Formatstring und die tatsächliche Eingabe, die in der Variablen 'inputstr' gespeichert ist, getrennt verwaltet werden.

Aus diesem Grunde werden auch die Cursorposition im Formatstring ('cur-

```
 1:  {
 2:   Diese Routine wurde geschrieben von:
 3:
 4:                  Bernhard Kohlhaas
 5:                  Talstrasse 17
 6:                  5130 Geilenkirchen
 7:
 8:
 9:
10:  }
11:
12:  program inputroutine;
13:
14:  {$r+} {meldet STRING-OVERFLOWS, die bei der Verwendung von 'concat'
15:         auf strings mit einer Laenge groesser 80 auftreten}
16:
17:  type chmenge = set of char; {notwendig fuer Inputroutine}
18:
19:  var s : string; {hier nur fuer Beispiele notwendig}
20:      ch : char;  { "     "    "        "         " }
21:
22:      ftaste :boolean; {notwendig fuer Inputroutine}
23:
24:  procedure clrscr;
25:    begin
26:      write(chr(27),'E')
27:    end;
28:
29:  procedure GotoXY(x,y:integer);
30:    begin
31:      if x<0
32:        then x:=0
33:        else if x>79
34:               then x:=79;
35:      if y<0
36:        then y:=0
37:        else if y>24
38:               then y:=24;
39:     write(chr(27),'Y',chr(y+32),chr(x+32))
40:    end;
41:
42:  procedure CursorAn;
43:    begin
44:      write(chr(27),'e')
45:    end;
46:
47:  procedure CursorAus;
48:    begin
49:      write(chr(27),'f')
50:    end;
51:
52:  function GetChar:char;
53:  const teiler = 65536;
54:  var erg           :long_integer;
55:      high,low      : integer;
56:
57:    function gchar:long_integer;gemdos(8);
58:
59:  begin
60:    erg:=gchar;
61:    high:=int(erg div teiler);
62:    low :=int(erg mod teiler);
63:    ftaste:=low=0;
64:    if ftaste then GetChar:=chr(high)
65:              else GetChar:=chr(low)
66:  end;
67:
68:  procedure input(xpos,ypos,maxlaenge :byte;
69:                  default        :string;
70:                  erlaubt        :chmenge;
71:                  Formatstring   :string;
72:                  var inputstr   :string);
73:
```

pos') und die tatsächliche Eingabeposition im Eingabestring ('strpos') getrennt verwaltet. Mit den Prozeduren 'CursorLinks' und 'CursorRechts' wird die neue Cursorposition in der Eingabemaske errechnet. Die tatsächliche Bewegung muß mit 'GoPos' bzw. 'Ausgabe' erfolgen. Außerdem muß das tatsächliche Verndern der Schreibposition im Eingabestring separat erfolgen. Die Prozedur 'CharRead' ist die Einleseroutine von der Tastatur, die Zeichen nur dann akzeptiert, wenn sie eine Kontrollsequenz sind (= ASCII 1–27 und 127), der letzte Tastendruck eine Funktionstaste war oder die Eingabe in dem durch 'erlaubt' vorgegebenen Buchstabenbereich liegt. In der Hauptroutine werden zunächst einmal verschiedene Initialisierungen vorgenommen. So wird der Überschreibmodus (Variable 'replace') ausgeschaltet, der Formatstring wird überprüft und gegebenenfalls an die tatsächlich erlaubte Länge der Eingabe angepaßt, dem Eingabestring wird der Defaultwert zugewiesen, dieser Wert wird ausgegeben und der Cursor an das Stringende gesetzt.

Die Eingabeschleife selbst besteht aus zwei Teilen, wobei der erste die Kontrollsequenzen und die Funktionstasten abarbeitet, der zweite die Zeicheneingaben erledigt. Die Kommandos und Operationen im ersten Teil sind praktisch selbsterklärend. Etwas sehr interessantes ist dabei anzumerken: Es war möglich, Tastaturcodes für die Funktionstasten und ASCII-Codes für die Kontrollsequenzen zusammen abzuarbeiten, da sich die Zahlenbereiche nicht überschneiden. So gibt z. B. bei der zweiten Option in der case-Schleife die Zahl 3 den ASCII-Code für Ctrl-C wieder und die 97 den Tastaturcode für UNDO.

Am Ende des Hauptprogramms sind noch einige Beispiele für die Verwendung dieser Routine gegeben, wobei mit diesen Beispielen schon die verschiedensten Anwendungen möglich sind, z. B. Datumseingabe, Eingabe eines Dateinamens oder Eingabe bzw. Ändern verschiedener Datensätze. Den Ideen sind hierbei sicherlich keine Grenzen gesetzt. Überlegen Sie doch selbst an welchen Stellen Ihrer eigenen Programme diese Routine einsetzbar wäre.

Viel Erfolg!

Bernhard Kohlhaas

```
 74:   {
 75:    Im Hauptdeklarationsteil sind folgende Typen
 76:    bzw. Variablen zu deklarieren:
 77:
 78:    type chmenge = set of char;
 79:
 80:    var ftaste : boolean;  Ist 'true', wenn letzter Tastendruck
 81:                           eine Funktionstaste war
 82:   }
 83:
 84:   const Standardformat='_';
 85:
 86:   {
 87:   Dieses Standardzeichen wird genommen, wenn im Formatstring
 88:   kein anderes gewaehlt wird.
 89:   }
 90:
 91:   var strpos,curpos,
 92:       i,nr,Formatlaenge   :byte;
 93:       eing,backgroundchar :char;
 94:       editierbar,fbereich :chmenge;
 95:       replace             :boolean;
 96:
 97:     procedure GoPos;
 98:     begin
 99:       GotoXY(xpos+curpos-1,ypos);
100:     end;
101:
102:     procedure Ausgabe;
103:     var i,stelle :byte;
104:
105:     begin
106:       CursorAus;
107:       GotoXY(xpos,ypos);
108:       i:=0;
109:       for stelle:=1 to length(inputstr) do
110:         begin
111:           i:=succ(i);
112:           if Formatstring[i] in editierbar
113:             then write(inputstr[stelle])
114:             else begin
115:                    repeat
116:                      write(Formatstring[i]);
117:                      i:=succ(i);
118:                    until Formatstring[i] in editierbar;
119:                    i:=pred(i);
120:                    stelle:=pred(stelle);
121:                  end;
122:         end;
123:         if i<Formatlaenge then write(copy(Formatstring,i+1,Formatlaenge-i));
124:         GoPos;
125:         CursorAn;
126:     end;
127:
128:     procedure Beep;
129:       begin
130:         write(chr(7))
131:       end;
132:
133:     procedure CursorRechts; {errechnet nur Cursorposition auf Bildschirm}
134:       begin                {erfordert Neuberechnung von 'strpos' sowie }
135:         repeat                         {Ausgabe mit GoPos oder 'Ausgabe'}
136:           curpos:=succ(curpos)
137:         until (curpos>length(inputstr)+nr)
138:            or (Formatstring[curpos] in editierbar)
139:       end;
140:
141:     procedure CursorLinks; {errechnet nur Cursorposition auf Bildschirm}
142:       begin                {erfordert Neuberechnung von 'strpos' sowie }
143:         repeat                         {Ausgabe mit GoPos oder 'Ausgabe'}
144:           curpos:=pred(curpos)
145:         until (curpos=1) or (Formatstring[curpos] in editierbar)
146:       end;
```

```
147:
148:     procedure Stringanfang; {bewegt Cursor zum Stringanfang}
149:       begin
150:         strpos:=1;
151:         curpos:=1;
152:         if not (Formatstring[1] in editierbar) then CursorRechts;
153:         ausgabe
154:       end;
155:
156:     procedure Stringende; {bewegt Cursor zum Stringende}
157:        begin
158:          Stringanfang;
159:          while strpos<=length(inputstr) do
160:            begin
161:              strpos:=succ(strpos);
162:              CursorRechts
163:            end;
164:          Ausgabe
165:        end;
166:
167:     function CharRead:char;
168:       var ch   :char;
169:           sound:boolean;
170:       begin
171:         sound:=false;
172:         repeat
173:           if sound then beep;
174:           sound:=true;
175:           ch:=GetChar;
176:         until (ch in (erlaubt+fbereich)) or ftaste;
177:         CharRead:=ch
178:       end;
179:
180:   begin {Routine Input}
181:     replace:=false;
182:     editierbar:=['_','$'];
183:     fbereich:=[chr(1)..chr(27),chr(127)];
184:     nr:=0;
185:     if length(Formatstring)=0 then Formatstring:=Standardformat;
186:     for i:=1 to length(Formatstring) do if not(Formatstring[i] in editierbar)
187:                                            then nr:=succ(nr);
188:     i:=1;
189:     loop
190:       exit if (Formatstring[i] in editierbar) or (i>length(Formatstring));
191:       i:=succ(i);
192:       if i>(maxlaenge+nr) then Formatstring:=concat(Formatstring,Standardformat);
193:     end;
194:     if i>length(Formatstring) then backgroundchar:=Standardformat
195:                             else backgroundchar:=Formatstring[i];
196:     while (maxlaenge>length(Formatstring)-nr)
197:       do Formatstring:=concat(Formatstring,backgroundchar);
198:     Formatlaenge:=length(Formatstring);
199:     inputstr:=default;
200:     Ausgabe;
201:     Stringende;
202:
203:         repeat
204:           eing:=CharRead;
205:           if ftaste or (eing in fbereich)
206:             then case ord(eing) of
207:
208:                    1 : {Ctrl-A: An Stringanfang gehen}
209:                          Stringanfang;
210:
211:                 3,97 : {Ctrl-C oder UNDO: Defaultstring wiederherstellen}
212:                          begin
213:                            inputstr:=default;
214:                            Stringende;
215:                          end;
216:
217:                    6 : {Ctrl-F: An Stringende gehen}
218:                          Stringende;
219:
220:                    8 : {Backspace: Zeichen links vom Cursor loeschen}
221:                          if strpos>1 then
222:                            begin
223:                              strpos:=pred(strpos);
224:                              delete(inputstr,strpos,1);
225:                              CursorLinks;
226:                              Ausgabe
227:                            end
228:                          else beep;
229:
230:                22,82 : {Ctrl-V oder Insert: schaltet Einfuegemodus um}
231:                         replace:=not replace;
232:
233:             25,27,71 : {Ctrl-Y oder ESC oder ClrHome: loescht Inputstring}
234:                         begin
235:                           inputstr:='';
236:                           Stringende
237:                         end;
238:
239:                19,75 : {Pfeil links oder Ctrl-S}
240:                         if strpos>1 then
241:                           begin
242:                             strpos:=pred(strpos);
243:                             CursorLinks;
244:                             Ausgabe;
245:                         end
246:                         else beep;
247:
248:                 4,77 : {Pfeil rechts oder Ctrl-D}
249:                         if strpos<=length(inputstr) then
250:                           begin
251:                             strpos:=succ(strpos);
252:                             CursorRechts;
253:                             ausgabe;
254:                           end
```

```
255:              else beep;
256:
257:         7,127 : {Delete Character under Cursor}
258:              if strpos<=length(inputstr) then
259:                begin
260:                  delete(inputstr,strpos,1);
261:                  ausgabe;
262:                end
263:              else beep;
264:
265:         end {case}
266:
267:     else begin
268:          if ( (eing in erlaubt) and
269:              (length(inputstr)<maxlaenge) or
270:              (replace and (strpos<=maxlaenge)) )
271:            then begin
272:              CursorAus;
273:              if strpos>length(inputstr)
274:                then inputstr:=concat(inputstr,eing)
275:                else if replace
276:                       then inputstr[strpos]:=eing
277:                       else insert(eing,inputstr,strpos);
278:              strpos:=succ(strpos);
279:              CursorRechts;
280:              Ausgabe;
281:            end
282:            else beep
283:        end;
284:   until eing=chr(13) {Abschluss der Eingabe mit RETURN}
285: end;
286:
287: {Im Hauptprogramm sind einige Beispiele fuer die Anwendung der
288:  Routine zu sehen.}
289:
290: begin
291:   clrscr;
292:   input(2,10,10,'',[' '..'z'],'_',s);
293:   clrscr;
294:   writeln;writeln;
295:   writeln(s);
296:   input(2,10,40,'Teststring',[' '..'z'],'',s);
297:   clrscr;
298:   writeln;writeln;
299:   writeln(s);
300:   input(3,10,6,'040486',['0'..'9'],'__.__.',s);
301:   clrscr;
302:   writeln;writeln;
303:   writeln(s);
304:   input(2,10,8,'TEST',['A'..'z','0'..'9'],'File: ________.TXT',s);
305:   clrscr;
306:   writeln;writeln;
307:   writeln(s);
308:   input(3,10,12,'A123bst4042n',[' '..'z'],'_/__/__-$$$$/',s);
309:   clrscr;
310:   writeln;writeln;
311:   writeln(s);
312: end.
```

Faites vos jeux
Atari auf der CES in Las Vegas



Auf dieser großen Elektronik-Messe wurden vom 8. bis zum 11. Januar 1987 die Neuigkeiten der Elektronik-Industrie vorgestellt. In der West Hall des Las Vegas Convention Center war diesmal die gesamte Computer-Branche untergebracht.

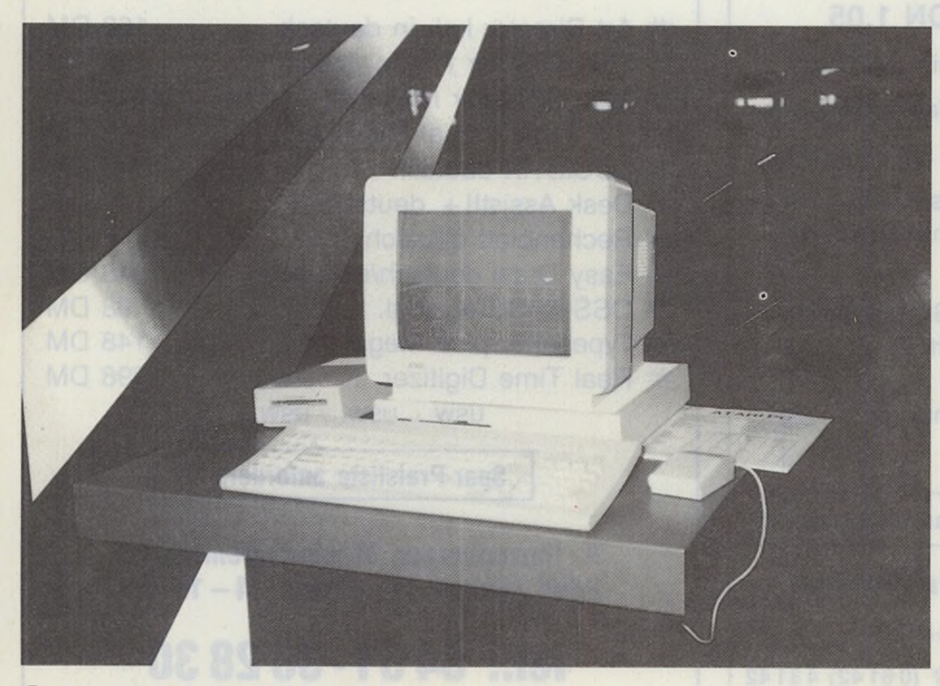
Commodore führte die in Braunschweig gefertigten IBM-kompatiblen Computer PC10 und PC20 auf dem amerikanischen Markt ein. Die Markteinführung des Amiga 2500 soll indes noch bis Ende Juni auf sich warten lassen. Trotzdem umlagerten treue Commodore-Fans den Stand und ließen sich interessiert die neuesten Softwareprodukte zeigen.

Atari wartete mit einer Sensation auf, mit der niemand gerechnet hatte: Auf dem farbenfrohen Stand wurde der staunenden Öffentlichkeit ein ATARI PC vorgestellt. Hier seine Leistungsdaten:

8080 CPU (8087 kann eingebaut werden), 8/4.77 MHz umschaltbar, 512 KB RAM ausbaufähig bis 640 KB, 256 KB Bildschirm-RAM (damit stehen die gesamten 512/640 KB des Hauptspeichers für Programme zur Verfügung), Darstellung im EGA-, CGA-, Monochrome- und Hercules-Mode mit einer maximalen Auflösung von 720 x 348 Punkten im monochromen Modus und 640 x 350 Punkten im farbigen Modus, parallel-, seriell und Maus-Port.

Der Atari PC kostet in den USA mit einem ausgezeichneten Monochrom-Monitor, Tastatur, einem eingebauten 5 1/4" Laufwerk und der Atari-Maus ganze 700 US-Dollar. Gegen Aufpreis gibt es auch einen Farbmonitor. Nach der Cebit 87 soll der Atari PC überall erhältlich sein.

In Deutschland soll der Preis, wie Alwin Stumpf, Geschäftsführer von Atari-Deutschland, auf Anfrage mitteilte, zwischen 1.800 und 2.000 DM liegen.



Der neue ATARI PC

Auf unsere Frage, wann Atari einen PC AT-kompatiblen Computer herausbringen werde, antwortete Stumpf lediglich, daß Atari nun auf zwei Computermärkten kräftig mitmischen wolle. Der Atari AT kommt also bestimmt!

Außerdem wurde der angekündigte Atari-Laserdrucker vorgestellt. Dieser Drucker kann nur an einem Atari ST mit mindestens 2 Megabyte Hauptspeicher betrieben werden, da die gesamte Steuerlogik per Software vom Atari ST kommt. Der Preis von ca. 3.000 DM ist eine Sensation. Rechnet man noch weitere 3.000 DM für einen Atari MEGA ST2 und ca. 300 DM für die passende Software hinzu, so kostet ein vollwertiges Desktop-Publishing-System weniger als 6.500 DM.

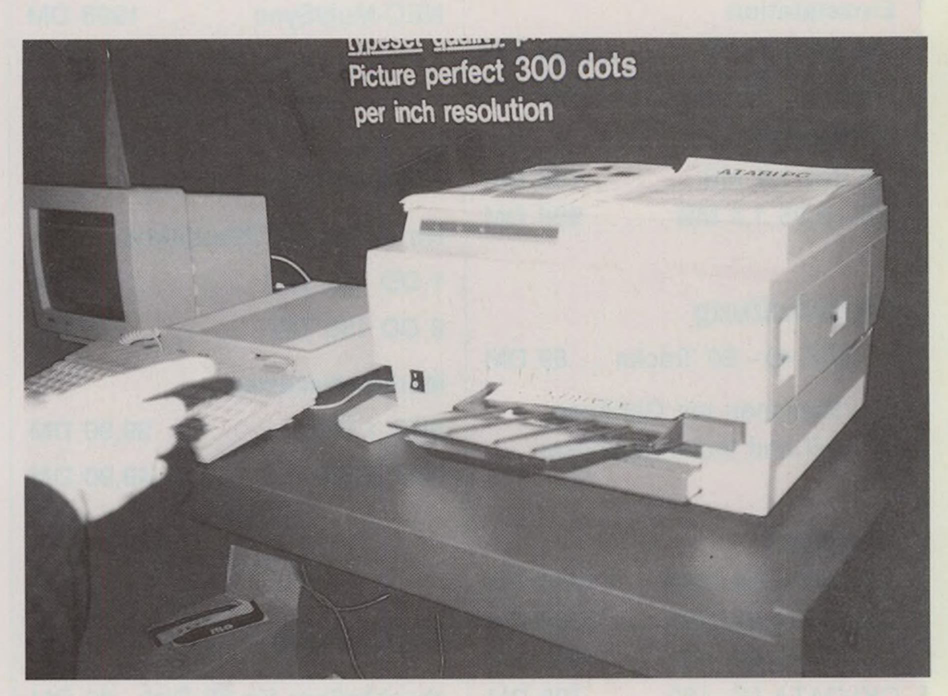


Bild 3: Laserdrucker für den ST

Erstmals offiziell vorgestellt wurden die Atari-Computer der MEGA-Serie. Diese neue Serie hat das Design eines „richtigen" Profi-Computers; die Tastatur ist also vom übrigen Computer getrennt. Das Computergehäuse, auf das der Monitor gestellt werden kann, hat ein eingebautes 3 1/2"-Laufwerk und ist wunderbar „slimline". Die Ataris der MEGA-Serie werden mit 1, 2 oder 4 MB Hauptspeicher geliefert, haben eine batteriegepufferte Uhr und den Blitter. Die neue 20 MB-Harddisk SH 205 hat das gleiche Gehäuse wie der MEGA-Atari, sodaß man beide Geräte wie HiFi-Komponenten übereinander stellen kann (s. Bild).

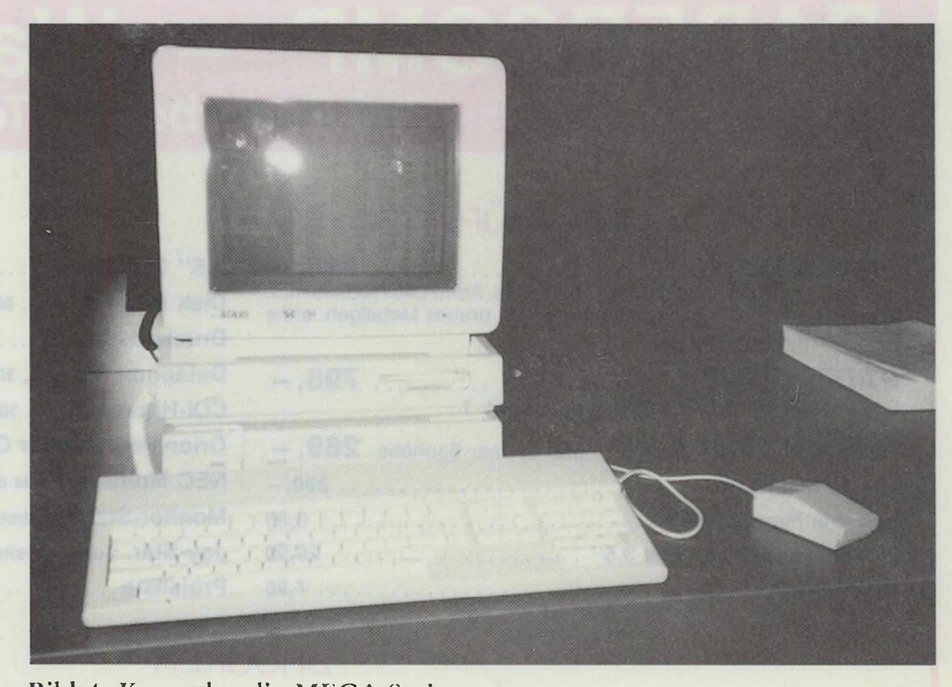


Bild 4: Kompakt: die MEGA-Serie

Atari kündigte eine Preissenkung für die Computer der ST-Serie um 20 Prozent an. Ob sie sich auch bei uns bemerkbar macht, wird sich zeigen.

Auch auf der Software-Seite hat sich einiges getan, so wurde Degas Elite, der Nachfolger von Degas, von Batteries Included vorgeführt.

GFA-BASIC, das bislang ca. 15.000 mal verkauft wurde, tritt nun endgültig seinen Siegeszug um die Welt an. Der amerikanische Exclusiv-Distributor MICHTRON führte vielen begeisterten Atari-Fans Interpreter und Compiler vor. Dabei war zu hören, daß GFA-Systemtechnik schon Software für den Atari PC fertig habe. So soll im März bereits GFA-DESK (Adress/Text/Mail, 99,– DM) und GFA-FAKT (Fakturierung/Artikel/Adress/Text/Mail, 149,– DM) auf dem Markt erscheinen. (RH)

Kleinanzeigen

BIETE HARDWARE

★ **Laufwerke, Gehäuse, Fest-** ★
★ **platten, Kabel und Stecker** ★
★ J – G Computertechnik ★
Unterdorfstr. 1 · 7850 Lörrach 8
★ 07621/52494 – 07622/63974 ★

■ DISKETTEN ■
■ 5 1/4", 48 tpi, DM 0,99, 2D ■
■ 3 1/2", 135 tpi, DM 3,19, 1DD ■
■ 3" Markendisk. DM 7,20 ■
■ auch andere, bes. Garantie ■
■ Allg. Austro-Ag. Ringstr. 10 ■
■ D-8057 Eching, T: 08133/6116 ■

SF314 DM 450; Tel. 091954863

SF 354 DM 150 0221/798436 18°°

520ST+ SF314 SM124 PCGeh+ Zubeh. neuw. 2200,- T. 02272/3320

SF-314 f. 375,– Tl: 0711/281541

SF354 160 DM 0251/23369 17 Uhr

Prommer Atari ST nur noch 165,– Kersting + Rehrmann 0561/777109, 61074, 283522

ST-FLOPPYSTATIONEN
★ Einzelstation 450,– ★
★ Doppelstation 780,– ★
in atarigrauen Metallgehäusen mit NEC 1036A. Sonst. Zubehör und Rabatte **Tel. 06201/22710**

520ST+, Monitor SM124, Doppelfloppy 2x720K, PC-Gehäuse mit abgesetzter Tastatur, ROM TOS, viele Programme + Literatur Tel. 08632/2155

Verk. Atari Farbmonitor SC1224 Suche monoch. SM124 02173/52249

Eprommer mit Textool nur 145,– **Accu-Uhr** wie St-Uhr nur 45 DM Uhr incl. Accus Tel. 040-862771

---

260 ST, 354 SF, TOS-ROMS, Maus NP ca 1300,-/VB 700,- Hitchhiker ST Tips & Tricks, PD Disks 1–18 aus ST Comp. zus. 200,- Tel. 0711/587673

Atari 1040STF + Monit. SM124 + Epson FX105 neuw. 2250 DM Tel. 0201/210801

SF354 DM 280. Tel. 05663/1558!

Atari ST Komplett-System incl. Software zu verkaufen. Preis n. VB Tel. 0821/812157

SF-314 DM 500,- und CSF PC-Geh. DM 100,- Weber Tel 0211/689313 ab 18°° 575366

Habe NEC P6 gewonnen, besitze leider keinen Computer. Daher verkaufe ich ihn. T: 07072/4354

3.5"-Floppy (720K), 4 Monate alt, 1/2 Jahr Garantie, kaum gebraucht, absolut neuwertig, anschlußfertig für ST: **450 DM!**
Arne, Tel. (02195) 2201 (Fr So)

SF354 neuw. 240 DM 04202/61289

SF354 neu orig. verp. DM 248,– Ab 18.30 h Tel. 06731/43884 VHB

260ST + SF354 + TOS-ROMS + Maus + ST Tips&Tr. + PD 1-18 + Hitchhikers G. Komplett 1000,– (NP ca. 1600,–) 0711/587673

Video-Digitizer-Pro=DM:450,-u. Maus-System m. kompl. Linsensatz =DM 80,-/Tel. 07843/1265 ab 18°°

★ **ST-Oszillograph** mit ★
★ **Sound Sampler** DM 399,– ★
★ **1MB Laufwerk** NEC 1036 A ★
★ DM 268,– ★
★ dto. Atari **modifiziert** ★
★ DM 298,– ★
★ Fl.-Gehäuse auf Anfrage ★
★ **Floppystation,** atarigrau ★
★ anschlußfertig DM 398,– ★
★ **Floppystecker** DM 11,90 ★
★ **anschlußf. Floppykabel** ★
★ ab DM 29,90 ★
★ Floppy-Kabel **mit inte-** ★
★ **griertem Treiber** DM 69,– ★
★ dto. für 2 Laufw. DM 89,– ★
★ **Netzteil 5V/12V** DM 39,90 ★
★ **PAL-mod.** f. Farb TV 275,– ★
★ **Speichererw.** steckbar ★
★ auf 1MB, auf Anfrage ★
★ **Epromkarte** 128KB DM 89,– ★
★ **Monitorstecker** DM 9,90 ★
★ **Monitorkabel** f. monochr. ★
★ Fremdmonitor DM 29,90 ★
★ **Scartkabel** 2m. DM 49,90 ★
★ **Druckerkabel** 2m. DM 29,90 ★
★ **Maus-Joystick,** Joystick ★
★ als Mausersatz, auf Anfr. ★
★ **Public D.-Progr. einzeln** ★
★ **zusammenstellen/** Liste ★
★ gegen DM 1,90 Rückporto ★
★ **Preise zuzügl. Versandkosten** ★
Nachn. 6,50 / Vorauskasse 4,50
Erfragen Sie Tagespreise
MIWIKO Computertechnik
Mesteroth 9 · 4250 Bottrop 2
Telefon: 0 20 45 / 8 16 38
oder ab 17 Uhr 0 20 43 / 7 13 74

Schachprogramm 'Deep Thought' Exklusiv bei uns, Preis a. A.
TV/AV-Modulatoren ab 130 DM
Ram-Erw. auf 1MB 225 DM
1x3.5"-Floppy für ST 455 DM
2x3.5"-Floppy für ST 865 DM
NEC 1036A (3.5"-Floppy) 278 DM
Floppy-/Monitorstecker uva.
Info bei: Stachowiak-Computer
4300 Essen, Burggrafenstr. 88
Tel.: 0201/273290

Schroeter Doppel-Floppy
3,5"/5,25" 1 Jahr alt 700 DM
Tel. 05121/65262

SF 354 11/86 VB 220,– DM
ab 17 Uhr Tel. 02421/62445

★ Verk. Epson RX 80: 490,– ★
Rolf Werdehausen, 6601 Klarenthal

Atari Mega-Anlage: Romtos, Taxan Drucker KP 810 St, 2 Floppy 354 über 100 Disketten, vieles mehr. Alles im Bestzustand. NP: 4200 DM VB: 3100 DM. Tel. 05327/2482

SF354 neuw. 200,– 0201/341457

Floppy-Modelle 3,5 Zoll:
ST- 3 -> 1★720 KB: 389 DM
ST-13 -> 2★720 KB: 699 DM
Modelle mit 5,25 Zoll:
ST- 5 -> 1★720 KB: 535 DM
ST-15 -> 2★720 KB: 965 DM
Modelle mit 3,5 und 5,25:
ST-16 -> 2★720 KB: 898 DM
Modelle mit 2★3,5 + 5,25:
ST-18 -> 3★720 KB: 1235 DM
★★ voll ATARI-Kompatibel
★★ mit NEC FD 1036A (3,5)
★★ komplett anschlußfertig
★★ eingeb. Netzteil
★★ 14 pol. Industriestecker
★★ mit oder ohne Gehäuse
>> NEC FD 1036A: 250 DM
–– alles Endpreise!! ––
Lieferzeit: jetzt nur noch
2–10 Tage (allg.)
FISCHER Computersysteme
Tel: 06151/51395

BIETE SOFTWARE

1st Word, DB Master One, Brataccas, Top Secret, je DM 50; Tel. 091954863

Orig.: Gfa-Basic-Compiler 130,– Cornerman (dt. Anleitung) 90,– Tel: 089/333452 oder 4485922

Wärmebedarfs./KZahlberechnung DIN4701/83 GEM-Bedienerführung Dipl. Ing. V. Koch, Am Mehnacker 11, 3563 Dautphetal 3, Tel 064687652

Baustatik-Software
Info anf. (1 DM) bei: Dipl. Ing. Jürgen Bullmann, Zur Hindenburgschleuse 3, 3000 Hannover 71

★★★ PD-SOFTWARE ★★★
Vielseitiges Angebot für ATARI ST (Anwendungen, Sprachen, etc) für jeden etwas, Liste gegen frank. Rückumschlag, H. Rode, Langreder Str. 1, 3000 Hannover 91

---

dBase II, Wordstar; neueste Version Datamat zu verkaufen; div. Bücher Tel: nach 18 h 07308-6461

Textomat, Text-Design, Datamat Karteikasten, alles m. Handb. u. Sicherungskopie. Zus. 200 DM bei: Reuter, Ringweg 28, 6920 Sinsheim

---

★ **Fernschreibprogramm DM 49** ★
★ Adressverwaltung/Terminverw. ★
★ sonstige Dateiverw. DM 35 ★
★ Martin M. Juilfs, 2943 Esens ★
★ Schützengärten 18 ★

Berlin: Preisverfall f. Public Domain! 70 Disks je DM 7,–, doppelseitig je 12,–. Freiinfo bei E. Behrendt, Kudowastr. 23A, 1000 Berlin 33, Tel. 030-8254163

Einkommensteuer/Lohnsteuer-Jahresausgleich jetzt 1986 u. 87
GfA/20 DM Vorkasse incl. Disk u. Porto / Jochen Höfer, Grunewald 2a 5272 Wipperfürth **02192/3368**

MIDI: Steinberg "24" (V1.1) 330 DM DX-7-**Voodoo** 260 DM, 030/8559848

★ **99 Pfennige** für **eine PD-Disk** ★
★ und weitere **Spitzenangebote** ★
★ für Soft- und Hardware!!! ★
★ Gratisinfo bei R. Markert Bal- ★
★ bachtalstr. 71 "Aktion 99" ★
★ 6970 Lauda-Oberbalbach ★

Gratis können Sie meinen Katalog anfordern und nicht viel teurer ist mein **riesiges, preiswertes und hochklassiges** PD-Angebot von über 140 Disks! R. Markert, Balbachtal 71, 6970 Lauda

---

Verkaufe meine Original-Programme **ART-** und **Film Director, Karate Kid II** und **GFA-Vektor.**
Tel.: 0931/76956

Verk. **Orig. GFA-Interp. V2.0** + "Das große GFA-Ba. Buch" (Data Be.) = 140 DM / Th. Leufkes 02368-55020!

PD-Software
Zusammenstellung der Disketten ist frei wählbar. Gratisinfo bei K. Galz, Sonnenbergstr. 55, 7800 Freiburg, Tel. 0761/60367

FORTRAN F77 (Prospero). Dialogboxen und Menüs. Beispiele auf Diskette. Vorauskasse DM 50,– Bossler, Germanenstr. 9, 7 Stuttgart 80

1st-Word und **Fußnotenprg.** umfngr. beide Orig. VB 128,– 0228/630629

Kleinanzeigen

GData (Util. + Ramd.): 50,–
1st_Word (dt.): 80,–
Psion: 50,– **Starglider:** 40,–
Tel.: 0431/674842 (Matthias)

GFA-Draft: 198,- / 1st-Word: 60,- GFA-Interpreter: 100,- / Copy-Star: 100,- / K-Switch: 50,- / DB-Mast.: 40,- / RAM-Disk-Lischka: 30,- / Profi-Painter: 60,- DM, alles orig. mit Handbuch. 02266/7894

ATARI-ST-Software in großer Auswahl ab Lager lieferbar.
Kostenlose Preisliste anfordern.
HEINICKE-ELECTRONIC
Kommenderiestr. 120
4500 Osnabrück
Tel. 05 41 5 8 27 99, Tx 944 966

Von Lehrer für Lehrer! Praxis-erpr. Notenverw.prg. voll **GEM** unterstützt mit **über 10 versch. Funkt.** (Info geg. fr. Rückumschl. 50 Pfg.) für DM 40,–! H. J. Mekel Nahestr. 28, 6600 Saarbrücken

---

★ **M. Pullen Software-Vertrieb**
Einführungsangebot:
Modula-2/ST 295,– DM
Modula Tools 148,– DM
Komplettpaket 430,– DM
02202/21784 (ab 18⁰⁰ Uhr)

Orig. **Protext** 99 DM 069/654476

ASPICE ein universelles Simulationsprogramm für Elektronik. Läuft auf **ST** mit **1MB:** Dipl. Ing. H. Ruff, Postf. 1942, 7910 Neu Ulm

---

★ ★ ★ **PD-SOFTWARE** ★ ★ ★
Leerdiskette: DM 3,90
Kopierkosten: DM 2,50
Info gratis, Edwar Twardoch
Kaiser-Wilh.-Str. 88, 1000 Berlin 46

PRINTMASTER Prg. text + Anleitung in Deutsch. Info: HAYOSOFT Flughafenstr. 46, 1000 Berlin 44

Vokabel-Prg. für den ATARI ST. GEM, bis zu 10 000 Vok., beidspr. Lernfkt., europ. Sonderzeichen, Suchen mit Wildcards, etc.
59 DM, Info 2 DM, Demo 10 DM
H. Michels, 3400 Gö, Föhrenweg 11

Einnahme-Überschuß-Rechnung
voll mausgesteuert **DM 99,–**
Heise-Software Tel: 05254/68581
4790 Paderborn, Simonstr. 17

SUPER-Graphikprogramm! Alle gängigen Funktionen. Vorkasse 69,-; GfA-Listing 15,– Rückporto beilegen Susanne Göhring, Katzwanger Hptstr. 26, 8500 Nürnberg

---

Lotto-Programm
nur noch 39 DM. Auf Wunsch 20seitige Dokumentation vorab.
Frank Klein, Dessauer STr. 5a
89 Augsburg 21, T. 0821/812157

VIP Professional (NP 750 DM)
DIGITIZER PRO (NP 900 DM)
Tobias WERNER n17⁰⁰ 07133/7017

---

db-Man, neue Version 2.02 G, nur DM 250,– 07151-18402

Matrizenalgebra in GFA Info: Fuchs, Traubenstr. 30, 7 Stuttgart 1

Atari ST – Wärmeschutznachweis DIN 4108 unter GEM, incl. Bauteile-Bibl.-Programm 420,– DM, mono, Demo-Disk 10,– DM, Born, Schlack 37, 4720 Kelmis/Belgien

---

Unterstützung für Einsteiger
Biete Anwenderprogramme für Handwerksbetriebe
Kalkul., Lohnbuchh. usw.
Großraum Stuttgart
Tel. (07024) 7767

Graphik-Artist 450,–; Easy-Draw 200,–; Mica-CAD 180,–; Degas 60,–; VIP-Profess. 450,–; d-BASEII 260,–; Wordstar 100,–; orig. + Handb. + Registerr.
Tel. 07843/1265 ab 18⁰⁰ Uhr!

GST-Assembler Orig. 80 DM, **ST-Pascal+** org. 190 DM 02831/8380

★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★ ★
★ **DATEX** Datenverwalt. (GEM) ★
★ GfA-Basic, dt. Umlaute, bel. ★
★ Ein-/Ausgabemasken, Such-/ ★
★ Sortierfkt., Summenbildung ★
★ ausführl. Anleitung, **DM 40,–** ★
★ **VIDEX** Videocass. verwaltung ★
★ Suchfkt., Filmverz., **DM 25,–** ★
★ **FIBEX** Finanzbuchhaltg. GEM ★
★ dopp. Buchung, Überschußber. ★
★ bis 1500 Buchungen, **DM 45,–** ★
★ Info T. 09421/33619 ab 17.00 ★

MABRU-SOFT: ARBEITSBLATT
Deskaccessory, Termin-, Kalender, Uhr Druck-Anpass. u. Notizbl. 40,– 10 3.5 Disk. 1SDD 50,– 2SDD 60,– **mit PD-Software** Thomas Brunner Buchenweg 15, 6301 Biebertal 7

PLOT_ST für Pro-FORTRAN! Einfachste Ausgabe grafischer Darstellungen auf Bildschirm und Drukker. Mehrere Funktionen gleichzeitig darstellbar! Komfortabel wie auf einem Großrechner. DM 95,-. Handbuch DM 15,- (wird verrechnet); Info gratis. Bestellung/Info bei:
M. Gamer, Friedrichsring 26,
6050 Offenbach

GEM-Applikationen (monochrom)
1) **KeyboardEdit:** zum Erstellen individueller Tastaturbelegungen, mit Installations-Accessory und AUTO-Lader **DM 32,–**
2) **RamSave, RamBoot:** superschneller Ramdisk-Blocklader (z. B. 600K in 40s) für Michtron-MDISK (ausschließlich) **DM 27,–**
3) **Topogramm:** dreidimensional-perspektivische Darstellung von Funktionsflächen, Zooming, Linienverd. ein/aus/neigungsschattiert, bel. Ansicht **DM 32,–**
4) **FormCalc:** programmierbarer argumentorient. Funktionen-Compiler /Interpr., 16 Funktionen je Satz, trigonometrische, hyperbolische, logische und andere Funktionen **DM 38,–**
5) **ASCII-Tabelle:** Accessory mit vollständigem Zeichensatz, Anzeige dez/hex/oktal und SDteuercodebedeutung **DM 18,–**
Info **GRATIS,** Versand per NN zzgl. Porto und **DM 5,–** f. Disk, (1S/2S angeben) Mattias Schick, Schänzle 22, 7500 Karlsruhe 21 Tel. 0721/752872

LOTTO-Prg, mausgest., GEM, s/w, 70k, DM 35,– Tel.: 04821/78629

ST-PD-Software! 1- und 2-seitig. 5,– DM inkl. Porto u. Diskette. Freikatalog gegen Rückumschlag. R. Schröer, Postlagernd, 46 Do 50

GFA-Basic Interpreter, ST-Pascal+ Flugsimulator II – alles original Tel. 0441-13606

SUCHE SOFTWARE

Suche, Habe, Tausche Atari ST Programme! Jede Menge! A. Lont Geuzenkade 75-3
1056KP Amsterdam

WERAPO-Aktienprogramm f. Atari
gesucht. Telef: 06104/63852

Suche **VIP** + Maus, **Adimens** und **Signum** preisgünstig.
Tel.: 07071-61542 ab 18 Uhr

Suche GfA-Basic Heimfinanz Haushaltsbuch usw. 0421/6039581

Suche Cobol Comp. 02291/6293

Lohn + Gehalt Texteverarb., CAD-CAM Wärmebedarf, sonstige aus Bau- u. Maschinenbau, Heizung, Klima u. Lüftung, Eck Roederichstr. 43, 6000 Frankfurt/M. 90 069-785729

Org. Megamax C; Tel. 091954863

KONTAKTE

ST-User su. **Erfahrungsaust.** MOSI HD/MA u. Umg. 06262/3206 ab 18h

---

Programmierer für Atari ST möglichst Lattice C gesucht
Tel: 04403-2167

Suche im Raum München Kontakt zu C-Programmierern 089/7241597

■■■■Tausche ST Software■■■■
■Liste an MARKUS BRUNOLD■
■Geisshofstieg3, 8200 Schaffhausen■
■ Schweiz, Tel. 00415341432 ■

VERSCHIEDENES

★ ★ **S T - T E R M I N +** ★ ★
Das ideale Terminverwaltungsprog. **für jeden ST-Besitzer!!!**
Sehr komfortabel; äußerst nützlich; **Kostl. Info anford.;** Anleitung vorab (5,– wird beim Kauf verrechnet) / 69,– / Th. Leufkes, Knappenstr. 10, 4353 Oer-Erkenschwick

---

Wer hat Erfahrung mit **Servicearbeiten** an PC- und ST-Computern? Interessante Verdienstmöglichkeit! Bitte anr. 08133/6163

Soft + Hardware für
ATARI 520ST, C-64, Ti-99/4A
MULTISOFT z. H. zillner Axel Loferer Bdstr. 10, A-5700 Zell

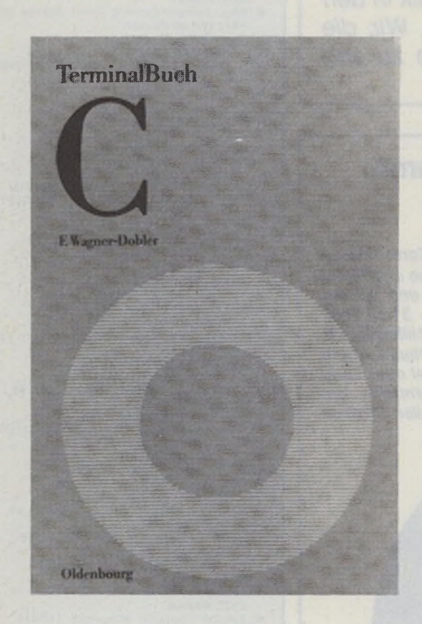
Brenne Programme und TOS auf Eprom. Tel: 05171/3723

SCHWEIZ – TIEFSTPREISE auf allen ATARI + AMIGA Computern, inkl. Peripherie + Software!!!
by COMPU-TRADE
Emmenstr. 16, CH-3415 Hasle-Rüegsau (034) 614593 auch abends

Verstehen Sie bei den Anleitungen zu englischer Software auch immer nur Bahnhof? Ihnen kann geholfen werden! Übersetze schnell, akkurat und preisgünstig!
Brockmann Translation Service Tel. 09366-6115

Sprachausgabe auf dem ST!
Info anfordern von:
Schlegel, Schwarzachstr. 3
7940 Riedlingen, 07371/2317

Kleinanzeigen
Privatanzeigen kosten pro Zeile und Spalte DM 7.– incl. MwSt.
Gewerbliche Kleinanzeigen kosten pro Zeile und Spalte DM 15.– zzgl. MwSt.
Zur Einsendung bitte die beiliegende Karte verwenden.
Bearbeitung nur gegen Vorausscheck (keine Überweisung).



F. Wagner-Dobler
TerminalBuch C
München 1985
R. Oldenburg Verlag
120 Seiten
DM 24,80
ISBM 3–486–29751–1

Das Buch liefert die preiswerte Beschreibung der Sprache für denjenigen, der von einer anderen Hochsprache auf C umsteigen will. Es ist ein Handbuch, das die Sprache lediglich beschreibt – und kein Programmierlehrbuch unter Verwendung von C. Wer Kniffe in dieser Sprache lernen will, der muß andere Bücher wie das „C-Puzzle Buch" kaufen.

Das „TerminalBuch" bezieht sich auf mehrere C-Compiler, unter anderem auch den DR- und den Lattice-Compiler, beide allerdings in der MS-DOS Version. Es ist jedoch anzunehmen, daß die auf dem ATARI ST verfügbaren den gleichen Sprachumfang haben. Bezieht sich eine Aussage nur auf einen Compiler, so wird dies extra vermerkt.

Dem Anspruch des Bandes entsprechend werden zunächst Sprachelemente, Datentypen, Operatoren etc. behandelt. Ein kleines Kapitel geht auf den Preprozessor ein. Schließlich wird noch eine Liste von Compilerbeschränkungen angegeben, die jedoch nur in seltenen Fällen von Nutzen sein wird.

Die zweite Hälfte des Buches beschäftigt sich mit den Funktionen aus den Standardbibliotheken. Die Auflistung muß natürlich nicht für jeden Compiler zutreffen. Die Güte eines Übersetzers läßt sich jedoch daran messen, wieweit er diese Bibliotheken zur Verfügung stellt, da sie der Kenighan & Ritchie-Sprachdefinition entsprechen.

Eine alphabetische Liste der Bibliotheksfunktionen und ein Register runden das Buch ab.

Laut Rückentext ist das TerminalBuch als „Begleiter neben dem Terminal" gedacht. Dieser Anspruch wird erfüllt. Der Band hält alle Informationen bereit, die man beim Programmieren nachschlagen muß.

Die Informationen sind detailliert, aber knapp formuliert. So werden etwa bei der Beschreibung der „scanf" Funktion alle möglichen Steuersequenzen komplett, aber trotzdem kompakt und übersichtlich dargestellt. Beispielprogramme darf man freilich nicht erwarten – sie widersprächen der Konzeption.

Wagner-Doblers C-Buch nützt also dem, der öfters mit C arbeitet, die Sprache aber nicht „auswendig" kann, und dem, der von einer anderen Hochsprache aus einen Abstecher nach C machen will. Der geringe Preis macht es dafür ideal.

Braun/Dittrich/Schramm
ATARI ST – Floppy und Harddisk
Düsseldorf 1986
Verlag Data Becker
522 Seiten

DM 59,–
ISBM 3–89011–132–7

Das Buch beschäftigt sich eingehend mit allen Interna, die Massenspeicher am ST betreffen. Es liefert zunächst eine Beschreibung des Zugriffs auf sequentielle und Random-Dateien von Basic, Pascal, Fortran (!) und C.

Das zweite Kapitel beschreibt die Datenstrukturen auf den Massenspeichern. Hier geht es etwa um Bootsektor, FAT, Programmheader oder Inhaltsverzeichnisse. Darauf folgt eine genaue Beschreibung des DMA-Controllers und dessen Verwendung mit Floppy oder Harddisk. Ein kurzes Beispiel beschäftigt sich auch mit RAM-Disks.

Dann folgt ein vollständig ausprogrammierter Disk-Monitor in Assembler, der direkten Zugriff auf jede Spur und die damit verbundenen Einzelheiten, wie Sync-Marken oder Gaps bietet (noch nicht gehört? Das sind eben die wirklichen Interna!). Das letzte Kapitel läßt auf recht geschickte Weise Basic-Programmierer auf den Floppy-Controller zugreifen.

Im Anhang finden sich die anscheinend unvermeidliche ASCII-Tabelle und ein Stichwortverzeichnis, das allerdings viel zu kurz geraten ist, als daß es eine Hilfe sein könnte. Außerdem bezieht es sich nur auf die Kapitelnummern und nicht auf Seiten.

Zu jedem Kapitel gibt es reichlich Beispiele in Assembler, die meist auch mit einem Basic-Lader für Leute ohne Assembler abgedruckt sind. Leider werden hier die Grenzen gesprengt: Wer will denn sage und schreibe 180 Seiten Assemblerlisting für den Diskmonitor abtippen? Oder die 26 Seiten Datas für den Basic-Lader?

Der Textteil des Buches ist fachlich kompetent und eine große Hilfe beim tieferen Einstieg in die Geheimnisse des ST. Er ist gut geschrieben sowie sinnvoll und reichhaltig illustriert. Viele Informationen sind noch nicht in andere Quellen zu finden. Das Hardcover erleichtert die Arbeit und läßt auf Haltbarkeit hoffen.

DM 59 DM erhält man eine ganze Menge an gut aufgemachten und hilfreichen Informationen. Die Diskette zum Buch mit Beispielprogrammen kostet 29 DM.

Robert Tolksdorf

Leserecke & Leserbriefe

NEC-Nadeldrucker P6
In Ihrem ST-Computerheft Nr. 12/86 bringen Sie auf Seite 58 einen Artikel über den NEC-Naeldrucker P6. In den letzten Tagen habe ich mir einen solchen Drucker gekauft und mußte feststellen, daß das Handbuch äußerst schlecht ist. Es wird z. B. mit keiner Silbe erwähnt, wie man an die vielen Schriftvarianten herankommt, die auch Sie in Ihrem Artikel erwähnen. Es werden zwar jede Menge Steuerzeichen angegeben, aber es wird nicht erklärt, wie man sie verwendet. Auch in Ihrem Artikel schreiben Sie nicht wie die Steuerzeichen verwendet werden. Was setzen Sie eigentlich voraus, wenn Sie einen solchen Artikel schreiben?
Wo finde ich ausführliche Literatur über die vielfältigen Möglichkeiten des NEC-P6 in Verbindung mit dem ATARI-ST?

Wolfgang Pfeiffer, Dormagen

Lassen Sie sich nicht entmutigen. Drucker-Handbücher sind prinzipiell nicht als Lehrbücher konzipiert, die verschiedenen Schriftarten sind jedoch vollständig erklärt. Wir glauben auch nicht, daß es Aufgabe eines Hardwaretests ist, die Bedienungsanleitung eines Produkts zu ersetzen. Inzwischen wird übrigens zum NEC-Pinwriter P6/P7 ein neues Handbuch ausgeliefert, das ziemlich alle Fragen beantworten dürfte. Käufer, die noch das alte Handbuch besitzen, können das neue Handbuch bei ihrem Händler erwerben.

Hardcopy-Quer-Listing
Das Hardcopy-Quer-Programm in Heft 1/87 funktioniert, wenn überhaupt, nur aus Zufall. Grund ist folgender: Wird ein Programm in den Speicher des ST geladen, werden seine absoluten Adressen so angepaßt, das es ab jeder Speicheradresse ablauffähig ist. Beim vorliegenden Assemblerlisting wird die Hardcopy-Routine hinter den Bildschirmspeicher verschoben, *nachdem* das Programm geladen wurde! Folglich behalten alle absoluten Adressen ihre ursprünglichen Werte!
Die Folgen können katastrophal sein. Jede Veränderung der Variablen doppelx, doppely und screen kann stattdessen gerade das geladene Programm, von dem aus die Hardcopyroutine aufgerufen wird (z. B. Textverarbeitung), zerstören und damit zum Absturz bringen.
Die Behauptung, das Programm ließe sich „problemlos" auf andere Assembler als den von DIGITAL RESEARCH übertragen, ist äußers gewagt. Der DR-Assembler optimiert nämlich alle absoluten Unterprogrammaufrufe via jsr automatisch zum programmzählerrelativen bsr, so dies möglich ist. Bei Assemblern, die das nicht tun, führt jsr aus den oben genannten Gründen zum Totalabsturz!
Wie kann man das alles vermeiden? Nun, theoretisch, wenn alle Speicherstellen programmzählerrelativ adressiert werden. Nur, einen Befehl wie move #1, doppelx(PC) kennt der MC 68000 gar nicht!
Tatsächlich kennt jedoch TOS eine sehr interessante Lösung für dieses Problem, nämlich den GEMDOS-CAL „keep", der das Programm resident im Speicher läßt. (Auf diese Art terminieren auch Ramdiskprogramme usw.)

Ich schlage vor:

```
begin:
pea patch
move #38, -(sp)          * Super
trap #14                 * XBIOS
addq.l #6,sp
clr -(sp)
move.l #ende-start+1000, -(sp)
move #$31, -(sp)         * keep
trap #1                  * GEMDOS
patch:
movea.l $456, a0         * _vblqueue
adda #28,a0
move.l #start, (a0)
rts
start:
tst $4ee ...
usw.
```

Noch zwei weitere Verbesserungsvorschläge:

1. Vor Label 10 sollte jsr groß stehen (sonst stimmt der Zeilenabstand für die erste Spalte nicht!)
2. Wenn es heißt
   graf__on: .dc.b 27,108, 10, 13, 27, 42, 5,144, 1,255
   und
   __exit: .dc.b 27,108, 0, 13, 10, 27, 65, 12,255

   dann wird die Hardcopy nicht so an den linken Rand gequetscht.

Bernhard Baier, Memmelsdorf

Themenwahl
Ich möchte mich der Meinung von Herrn Braun aus Gladbeck, dessen Leserbrief in der Januar-Ausgabe veröffentlicht wurde, anschließen.
Nach meinen Erfahrungen programmieren höchstens 25 % aller ST-Besitzer selbst, die meisten davon in Basic. Der Rest verwendet den Computer hauptsächlich zur Verarbeitung von Texten, Daten oder Grafiken und nicht zuletzt zur Unterhaltung. Diesen Ansprüchen wird die Themenzusammensetzung in der ST-Computer nicht gerecht. Zu oft gibt es mehrseitige Listings in C oder Pascal sowie längere Kurse über die Programmierung des Betriebssystems. Während einfachere Sprachen wie beispielsweise GFA-Basic vernachlässigt werden, ziehen sich Tests von recht exotischen Produkten (s. 1/87: LISP) häufig zu sehr in die Länge.
Meiner Ansicht nach sollte man sich noch weiter auf die Vorstellung neuer Anwender-Software sowie die Veröffentlichung nützlicher „Tips & Tricks" konzentrieren.
Ein weiterer Kritikpunkt ist die mangelnde Aktualität Ihrer Artikel. Häufig kann man in Zeitschriften, die sich nicht allein auf den ST spezialisiert haben, schon zwei oder drei Monate vorher Berichte über Neuvorstellungen oder andere Themen lesen. Das sollte sich ändern.

Ulf Blanke, Brome-Altendorf

In Zukunft wollen wir auch Basic wieder in höherem Maße berücksichtigen. Wegen der Fehlerhäufigkeit des ST-Basic (darüber wurde ja bereits vielfach berichtet) verzichten wir lediglich auf diesen Interpreter.
Tatsächlich ist der LISP-Bericht etwas länger ausgefallen. Wir wollen jedoch unseren Lesern 'exotische' Sprachen deshalb nicht vorenthalten, weil gerade die KI-Sprachen in Zukunft voraussichtlich große Bedeutung erlangen werden.
Die von Ihnen angesprochenen Tips & Tricks werden Sie ab dieser Ausgabe zum Teil in der ST-ECKE finden. Wir hoffen, damit einer großen Anzahl von Lesern gerecht zu werden, die mit Fragen auf uns zukommen. Auch Hardware-Tips werden wir weiterhin in unregelmäßigen Abständen veröffentlichen.
In punkto Aktualität versuchen wir allerdings, unter den ersten zu sein. Es gibt nur eines zu bedenken: Wir testen grundsätzlich keine Vorabversionen (siehe Editorial 2/87), da damit keine zuverlässige Aussage über die Qualität eines Produktes getroffen werden kann. Wenn man von diesem Vorsatz abrückt, kann man ohne Probleme einige Monate früher berichten. Am Beispiel von 1st__Word hat sich gezeigt, daß zwischen den ersten Pseudo-Tests einiger Zeitschriften und dem wirklichen Erscheinen des Produkts über ein halbes Jahr vergangen ist.

Bitte beantworten Sie mir als Computerneuling eine möglicherweise 'dumme Frage': Handelt es sich bei den im Disketten- und Public-Domain-Service angebotenen Programmen durchweg um auf jedem ST-Rechner lauffähige Versionen oder sind zum Betrieb der Programme bestimmte Programmiersprachen erforderlich?

Wolfgang Preisler, München

Bei Einstieg in eine neue Materie gibt es grundsätzlich viele Fragen. Sie sollten sich nicht scheuen, diese auch zu stellen. Um Sie zu beruhigen: Sämtliche Programme sind lauffähig. Falls dazu Hilfsprogramme erforderlich sind (z. B. der GFA-Run-Only-Interpreter), sind diese selbstverständlich beigefügt.

Leser antworten Lesern

In dieser Rubrik können Sie, liebe Leser, unsere Zeitschrift mitgestalten. Jeder, der Lust und Kenntnis hat, kann seine Frage oder Antwort an uns schicken. Wir werden sie dann veröffentlichen.

Ich habe auf einer Diskette ein 45K langes Basic-Adventure gespeichert, das ich gerade entwickle. Doch auf ein Mal spielte die Diskette verrückt: Immer, wenn ich das Directory laden will (auf der Diskette sind nur drei Ordner, die voll mit PRGs sind) sehe ich die drei Ordner und die Angabe „400000000... Bytes belegt". Dieses Bild ist einige Sekunden auf dem Bildschirm, dann erscheinen weiße senkrechte Streifen (schwarz auf weißem Grund), die jeweils einen Abstand von etwa 10 Pixeln zueinander haben. Dann verabschiedet sich der Computer. Ich biete dem DM 10,– der es schafft, mein Programm wieder zum laufen zu bringen, denn anscheinend ist ja nur das Directory kaputt. Meldet euch bitte bei: Carsten Lutz, Königsberger Str. 17, 2084 Rellingen 1

Carsten Lutz, 2084 Rellingen 1

Als Anwender der Kalkulationsprogramme VIP auf Atari und Lotus 1-2-3 auf IBM-AT habe ich folgende Fragen: Lassen sich auf 1-2-3 erstellte Arbeitsblätter mit Vip weiterverarbeiten? Wenn ja, mit welchem Laufwerk kann man unter MS-DOS-Format laufende Files mit VIP weiterverarbeiten? Und umgekehrt, lassen sich auf Vip erstellte Templates auf MS-DOS-Rechnern weiterverarbeiten? Für Tips wäre ich sehr dankbar. Welche Soft- und Hardware (5 1/4 Zoll-Laufwerke) wird benötigt?

H. D. Schultz, 7140 Ludwigsburg

Druckeranpassung
für TA Gabriele 9009 (1__Word)
Sehr geehrte Damen und Herren,
seit etwa 4 Monaten betreibe ich die Gabriele 9009 an meinem Atari ST zusammen mit dem Textverarbeitungsprogramm 1__Word.
Leider habe ich einige Probleme bei der Einzelblattverarbeitung. Möchte ich mehrere Blätter hintereinander ausdrucken, passiert folgendes: Das erste Blatt wird korrekt gedruckt, der Rechner meldet sich und fordert zum Papierwechsel auf. Habe ich nun ein neues Blatt eingespannt und bestätige dies, dann erfolgt beim Drucken kein Zeilenvorschub mehr. Alle Zeilen werden also übereinandergedruckt!
Ich arbeite mit der Druckeranpassung aus der Septemberausgabe der ST-Computer und habe die Version 1.06 (BRD) von 1__St Word.
Ich würde mich freuen wenn Sie mir bei meinem Problem behilflich sein könnten.

Andreas Ohlmann, 6694 Marpingen

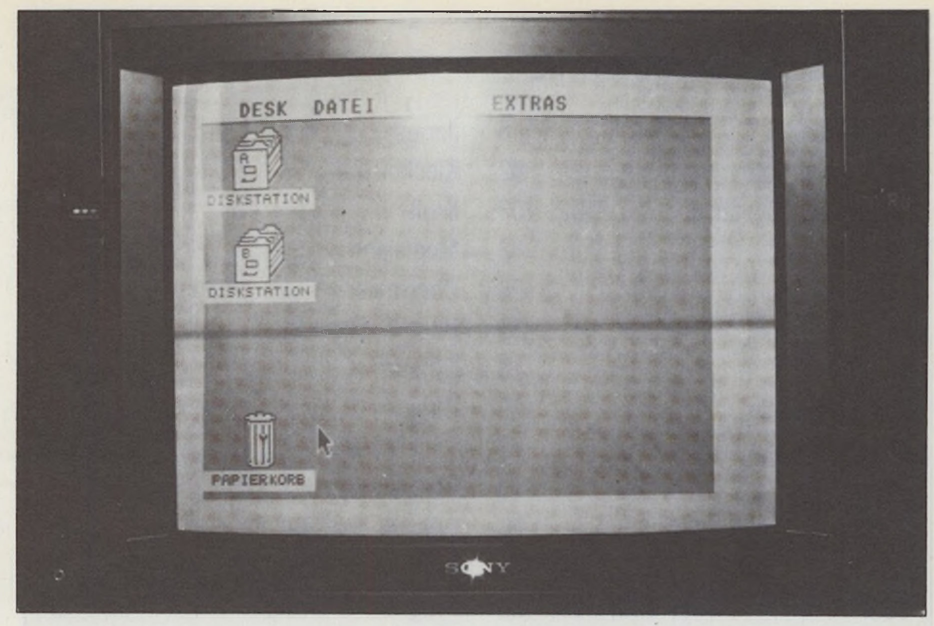
VDISYS mit Macken?
Nach dem Aufruf Setwritemode (VDI-SYS 32) beliebt mein ST nach Druck der RETURN-Taste (im Direktmodus) abzustürzen. Wie kann diesem Übel abgeholfen werden?

Christian Pollmann, 8481 K'naab

In der nächsten



lesen Sie unter anderem



Farbfernseher am ST

Mit SCART-Buchsen versehene Fernsehgeräte können problemlso an den ST angeschlossen werden. Doch wie gut ist das Bild? Besser oder schlechter als das eines waschechten Computermonitors? Der Test ermittelte verblüffende Ergebnisse.

Speicherzuwachs im EPROM

Die Eprombank für unseren EASY-PROMMER ist nach einigen Versuchen dem Entwicklungslabor entkommen. Sie hat eine Speicherkapazität von 512 KByte und kann dadurch gleich mehrere Programme aufnehmen. Die meistbenutzte Software ist somit resident im Rechner und per Mausklick sofort vorhanden. Im nächsten Heft werden wir diese Eprombank endlich vorstellen.

ST-Ecke

In dieser Folge wird beispielsweise das Geheimnis der Joystickabfrage gelöst und zwar auf saubere Art und Weise, was sehr wichtig ist. Lassen Sie sich überraschen.

Heft 4/87 erscheint am 27. März 1987

Änderungen vorbehalten

Zwei Video-Digitizer im ST-Testlabor

Der Draht zur Außenwelt war schon immer interessant. Ein Bild aus einer Videokamera in den Rechner zu befördern, ist einer dieser 'Drähte' – und ein besonders heißer dazu. Doch die Leistungsfähigkeit der Digitizer ist unterschiedlich. Wir testen neue Geräte.

1st-WORD PLUS

Lange ist das beliebte Textprogramm in tiefen Kellern gereift und hat auch deutlich an Geschmack zugenommen. In knappen Worten: Grafikeinbindung, deutsches Lexikon, Fußnotenverwaltung, Serienbrieferstellung und Silbentrennung. Ein Bericht aus unserer Testküche.



Inserentenverzeichnis

ST-Computer – Einzelheft-Bestellung

ST-COMPUTER können Sie direkt beim HEIM-VERLAG zum Einzelheft-Preis von DM 6,– (zuzüglich Gebühr für Porto u. Verp.) nachbestellen.
Bearbeitung **nur gegen beigefügten Scheck** über den entsprechenden Betrag (keine Überweisung).

Jan.	Febr.	März	April	Mai	Juni	Juli/Aug.	Sept.	Okt.	Nov.	Dez.	
		–––									1986 = DM
											1987 = DM

+ Gebühr für Porto u. Verp. = DM

☐ Scheck in Höhe zus. DM ______ liegt bei

Vorname/Name______

Straße/Hausnr.______

Ort______

Datum/Unterschrift______

Gebühr für Porto u. Verpackung: 1 Heft DM 2,–; 2 bis 6 Hefte DM 3,–; ab 7 Hefte DM 5,–

Heim-Verlag · Heidelberger Landstraße 194 · 6100 Darmstadt-Eberstadt

Impressum

ST-Computer

Herausgeber:
Heim Fachverlag,
Heidelberger Landstraße 194,
6100 Darmstadt 13,
Telefon (0 61 51) 5 60 57

Verlagsleitung: Hans-Jörg Heim

Redaktion:
Uwe Bärtels (UB Chefredakteur)
Marcelo Merino (MM)
Harald Egel (HE)
Dipl.-Ing. Harald Schneider (HS)

ST-Computer
Redaktion
Schwalbacher Str. 64
6236 Eschborn
Tel. 0 61 96/48 21 58
oder
Postfach 59 69, 6236 Eschborn

Redaktionelle Mitarbeiter:
Markus Nerding (MN)
Oliver Joppich (OJ)
Uli Eickmann (UE)
Stefan Höhn (SH)
Jürgen Leonhard (JL)
Chr. Schormann (CS)
Jörg Wilhelm (JW)

Autoren dieser Ausgabe:
Sven Schuler
Dr. Volker Kurz
Rolf Hilchner
Dr. Karl Sarnow
Robert Tolksdorf
Klaus Heuer
Peter Monadjemi
Bernhard Kohlhaas

Produktion:
Klaus Schultheis (Ltg.),
Bernd Failer
Susanne Failer
Bela Kumar
Carola Schwarze
Martina Simacek

Anzeigenverkaufsleitung: Uwe Heim

Anzeigenpreise:
nach Preisliste Nr. 2, gültig ab 1.7.86

Vertrieb: Hans-Jörg Heim, Uwe Heim, Heide Schultheis

Erscheinungsweise: 11 x jährlich

Bezugspreis: Einzelheft DM 6,–. Jahresabonnement DM 60,– inklusive der gesetzlichen Mehrwertsteuer und den Zustellgebühren für 11 Ausgaben.
Ausland: 80,– DM inkl. Versand

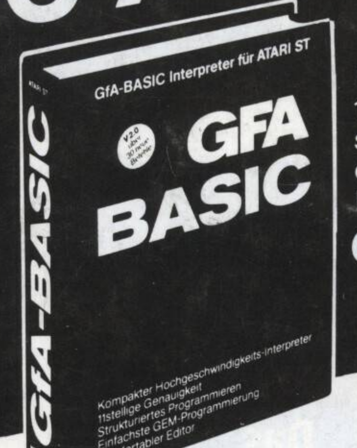
Bezugsmöglichkeiten: ATARI-Fachhändler, Zeitschriftenhandel, Kauf- und Warenhäuser oder direkt beim Verlag unter obiger Adresse.

Druck: Ferling Druck Darmstadt

Manuskripteinsendungen: Programmlistings, Bauanleitungen und Manuskripte werden von der Redaktion gerne angenommen. Sie müssen frei von Rechten Dritter sein. Mit ihrer Einsendung gibt der Verfasser die Zustimmung zum Abdruck und der Vervielfältigung auf Datenträgern im Heim Verlag. Honorare nach Vereinbarung. Für unverlangt eingesandte Manuskripte wird keine Haftung übernommen.

Titelseite: Fabian & Mayer
Foto: Rainer Spirandelli

Für alle ATARI ST



Kompakter Hochgeschwindigkeits-Interpreter,
11stellige Genauigkeit,
strukturiertes Programmieren,
einfachste GEM-Programmierung,
komfortabler Editor.

GFA-BASIC Interpreter V 2.0
DM 169,–

GFA-BASIC Compiler für ATARI ST
GFA
BASIC
GFA-BASIC

GFA-BASIC Compiler
DM 169,–

Schneller 2-Pass-Compiler, benötigt keinen Linker,
erzeugt sehr schnelle, kurze Programme ohne Runtime-Modul,
voll kompatibel zum Interpreter, einfachste Bedienung.

GFA-Buch ATARI ST
GFA
BASIC
GFA-BASIC
Frank Ostrowski

GFA-Buch DM 79,–

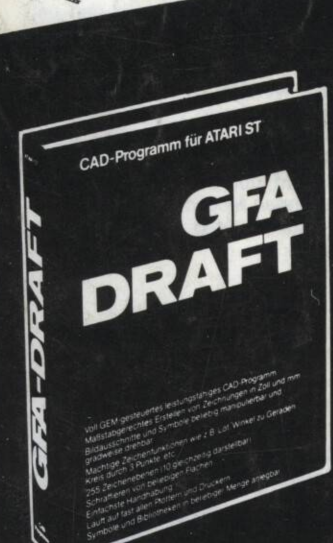
Incl. Diskette mit Beispiel-Programmen
Lieferbar voraussichtlich Ende Februar 87



Schnelles, interaktives 3D-Grafik-Programm zum Generieren von 2D- und 3D-Objekten.

Aus GFA-BASIC heraus können problemlos 2D- oder 3D-Spiele, Animationen oder bewegte Simulationen erzeugt werden.

GFA-VEKTOR DM 149,–



Voll GEM-gesteuertes, leistungsfähiges CAD-Programm, maßstabsgerechtes Erstellen von Zeichnungen in Zoll und mm, Bildausschnitte und Symbole beliebig manipulierbar und gradweise drehbar, mächtige Zeichenfunktionen wie z. B. Lot, Winkel zu Geraden, Kreis durch 3 Punkte, etc. 255 Zeichenebenen (10 gleichzeitig darstellbar). Schraffieren von beliebigen Flächen. Einfachste Handhabung. Läuft auf fast allen Plottern und Druckern. Symbole und Bibliotheken in beliebiger Menge anlegbar.

GFA-DRAFT DM 298,–

...Anruf genügt: 02 11-58 80 11

GFA Systemtechnik GmbH

GFA SYSTEMTECHNIK

Heerdter Sandberg 30
D-4000 Düsseldorf 11
Telefon 02 11/58 80 11